NAVAL POSTGRADUATE SCHOOL

Monterey, California

THESIS

APPLICABILITY OF COMPUTER SPREADSHEET SIMULATION FOR SOLVING RESOURCE ALLOCATION PROBLEMS

by

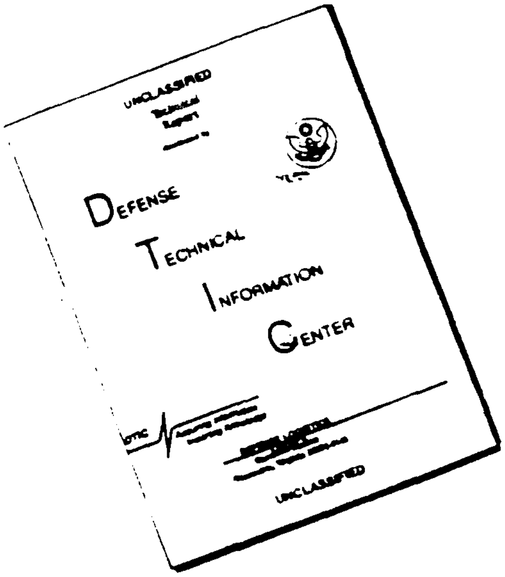
Dixon Kendall Hicks

March 1993

Principal Advisor Shu S. Liao

Approved for public release; distribution is unlimited

DISCLAIMER NOTICE



UNCLASSIFIED

Technical Report

DEFENSE TECHNICAL INFORMATION CENTER

DTIC

UNCLASSIFIED

THIS DOCUMENT IS BEST QUALITY AVAILABLE. THE COPY FURNISHED TO DTIC CONTAINED A SIGNIFICANT NUMBER OF PAGES WHICH DO NOT REPRODUCE LEGIBLY.

Unclassified
SECURITY CLASSIFICATION OF THIS PAGE

REPORT DOCUMENTATION PAGE			*Form Approved* *OMB No. 0704-0188*
1a REPORT SECURITY CLASSIFICATION: UNCLASSIFIED		1b RESTRICTIVE MARKINGS	
2a SECURITY CLASSIFICATION AUTHORITY		3 DISTRIBUTION/AVAILABILITY OF REPORT Approved for public release; distribution is unlimited.	
2b DECLASSIFICATION/DOWNGRADING SCHEDULE			
4 PERFORMING ORGANIZATION REPORT NUMBER(S)		5 MONITORING ORGANIZATION REPORT NUMBER(S)	
6a NAME OF PERFORMING ORGANIZATION Naval Postgraduate School	6b OFFICE SYMBOL *(if applicable)* 36	7a NAME OF MONITORING ORGANIZATION Naval Postgraduate School	
6c ADDRESS *(City, State, and ZIP Code)* Monterey CA 93943-5002		7b ADDRESS *(City, State, and ZIP Code))* Monterey CA 93943-5002	
8a NAME OF FUNDING/SPONSORING ORGANIZATION	8b OFFICE SYMBOL *(if applicable)*	9 PROCUREMENT INSTRUMENT IDENTIFICATION NUMBER	
8c ADDRESS *(City, State, and ZIP Code)*		10 SOURCE OF FUNDING NUMBERS	

PROGRAM ELEMENT NO	PROJECT NO	TASK NO	WORK UNIT ACCESSION NO

11 TITLE *(Include security classification)* APPLICABILITY OF COMPUTER SPREADSHEET SIMULATION FOR SOLVING RESOURCE ALLOCATION PROBLEMS

12 PERSONAL AUTHOR(S) Dixon K. Hicks

13a TYPE OF REPORT Master's Thesis	13b TIME COVERED FROM___TO___	14 DATE OF REPORT *(YEAR, MONTH, DAY)* 1993, MARCH	15 PAGE COUNT 140

16 SUPPLEMENTARY NOTATION The views expressed in this thesis are those of the author and do not reflect the official policy or position of the Department of Defense or the U.S. Government.

17 COSATI CODES			18 SUBJECT TERMS *(Continue on reverse if necessary and identify by block number)*
FIELD	GROUP	SUBGROUP	Spreadsheet Simulation, Resource Allocation, Inventory Distribution Management, Queuing Utilization, Financial Budgeting, Simulation

19 ABSTRACT *(continue on reverse if necessary and identify by block number)*

This thesis investigates the possibility of employing computer spreadsheets as a sophisticated tool to resolve resource allocation dilemmas through simulation techniques. *Microsoft's Excel 4.0* is used to analyzed three separate and unique resource allocation problems. First, an inventory distribution system involving different distribution points to illustrate the magnification of uncertainty as the distribution system is lengthened. Second, queuing utilization problem faced by an emergency room of a hospital. The third scenario looks at the uncertainty in financial budgeting situation as reflected in the Navy's CHAMPUS budget. A spreadsheet macro using simulation techniques is created for each scenario to illustrate that computer spreadsheets are fully capable of analyzing resource allocation enigmas through simulation methodology.

20 DISTRIBUTION/AVAILABILITY OF ABSTRACT ☒ UNCLASSIFIED/UNLIMITED ☐ SAME AS REPORT ☐ DTIC USERS		21 ABSTRACT SECURITY CLASSIFICATION UNCLASSIFIED	
22a NAME OF RESPONSIBLE INDIVIDUAL Professor Shu S. Liao		22b TELEPHONE *(Include Area Code)* (408) 656-2505	22c OFFICE SYMBOL AS / Lc

Previous editions are obsolete

S/N 0102-LF-014-6603

SECURITY CLASSIFICATION OF THIS PAGE
Unclassified

Applicability of Computer Spreadsheet Simulation for Solving Resource Allocation Problems

by

Dixon Kendall Hicks
Lieutenant, United States Navy
B., Georgia Institute of Technology, 1985

Submitted in partial fulfillment of the
requirement for the degree of

MASTER OF SCIENCE IN MANAGEMENT

from the

NAVAL POSTGRADUATE SCHOOL
March 1993

Author: ______________________________
Dixon Kendall Hicks

Approved by: ______________________________
Shu S. Liao, Principal Advisor

____
RADM Richard D. Milligan, USN (Ret), Associate-advisor

____
Davis Whipple, Chairman, Department of Administrative Science

ABSTRACT

This thesis investigates the possibility of employing computer spreadsheets as a sophisticated tool to resolve resource allocation dilemmas through simulation techniques. *Microsoft's Excel 4.0* is used to analyzed three separate and unique resource allocation problems. First, an inventory distribution system involving different distribution points to illustrate the magnification of uncertainty as the distribution system is lengthened. Second, queuing utilization problem faced by an emergency room of a hospital. The third scenario looks at the uncertainty in financial budgeting situation as reflected in the Navy's CHAMPUS budget. A spreadsheet macro using simulation techniques is created for each scenario to illustrate that computer spreadsheets are fully capable of analyzing resource allocation enigmas through simulation methodology.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF ILLUSTRATIONS

I. INTRODUCTION

Within the Department of Defense and business world, one facet of a manager's success is his/her skill at formulating an appropriate balance among the scarce resources at his/her disposal. Sagacious allocation of limited resources are paramount to the success of a manager. Multiple methods exist and are at the disposal of a manager for guiding him/her to the most feasible solution. One of the instruments available to the manager for resolving a resource allocation dilemma is simulation. Simulation is a methodology that handles uncertainty and presents the manager with the best possible solution among several feasible alternatives. The most simplistic simulation model involves elementary mathematical equations and events that can be formulated and solved by hand. However, most applications of simulation in a real world setting are too complex for hand calculations and thus require the implementation of digital computers with simulation specific software. The simulation method that is appropriate for each predicament is a function of the complexity of the problem and the time constraints faced by the decision maker. Hand simulation is time consuming and often impossible to solve. Simulation specific packages employing a digital computer will solve a majority of problems. Unfortunately, simulation oriented software is not as widely applied as it can be due its prohibitive cost or technical skill needed to develop a simulation program.

A. PURPOSE

The purpose of this thesis is to confront resource allocation through simulation methodology by using a conglomeration of simplistic and complex methods. Digital computer spreadsheets, which are available to virtually every manager, can be applied to perform simulation. A link between computer spreadsheets and simulation will allow a broader application of simulation methodology by managers. This thesis will focus on the methods for applying computer spreadsheet simulation for solving relatively complex resource allocation predicaments. This study is structured to answer one primary question: How applicable are off-the-shelf digital computer spreadsheets for resolving resource allocation problems employing simulation methodology?

B. SCOPE

The focus of the study will be limited to discrete vice continuous simulation techniques. The distinction between discrete and continuous systems is required because of the entirely separate discipline existing concerning the study of continuous systems. Models of continuous systems are an industrial process integrated over a period of time resulting in the mathematical formulas containing differential equations. Discrete systems involve product industries that can be quantified into discrete events thus requiring simplistic mathematical equations. Problems requiring the application of differential equations should be solved with simulation specific software. Thus, the analysis will be conducted using the digital computer spreadsheet software *Microsoft Excel 4.0 for Windows*. Excel 4.0 is currently the most powerful off-the-shelf spreadsheet software available and is compatible with other available spreadsheet software

C. OVERVIEW

In order to comprehend the simulation method, simulation philosophy will be introduced in Chapter II. The chapter will also describe how simulation can be used as a resource allocation tool. Chapter III introduces terms, builds upon the simulation philosophy, and develops the methodology required for construction of simulation models. Throughout the chapter, simulation methodology as it pertains to computer spreadsheets will be discussed in order to develop guidelines for building models. Chapters IV, V, and

VI will illustrate the previous chapter's guidelines for computer spreadsheets. Three different scenarios will be modeled and analyzed using simulation. Chapter VII presents the conclusions of the research.

II. SIMULATION AS A RESOURCE ALLOCATION DECISION TOOL

Simulation methodology is one of several tools available to the manager for providing feasible solutions to the enigma of allocating scarce resources. Before discussing the many benefits of analyzing resource allocation problems through simulation techniques, one must first become familiar with the alternative methods that are available and practiced by managers. When presented with a predicament concerning the apportionment of assets, how does a manager arrive at a decision?

A. TOOLS FOR RESOURCE ALLOCATION

A significant number of the decisions made by managers when facing almost any issue are founded upon his/her previous experience or intuition. A manager's experience accumulates throughout his/her career and can originate from many sources. The acumen of professional consultants, professional literature, and successes or failures both he/she and his/her competitors have encounter in the past are just a few of the sources of experience. Decisions arrived at by intuition are more difficult to rationalize. However, a decision based on a "gut feel" has been encountered by almost everyone. Recurrently, a manager encounters a decision that he/she has no experience to reflect upon for a solution. What methods are available to a manger if he/she has no previous experience or no desire to commit resources solely upon intuition?

One method a manager has at his/her disposal if he/she does not have any experience to reflect upon is to create a knowledge base for his/her decision. Knowledge is developed by performing experiments on the actual system that he/she lacks knowledge and observing the responsive behavior. The system that is modeled can be any set of interdependent elements that function within an organization to meet specific goals, i.e., allocation of resources. Experiments consist of proposing and applying changes to variables, policies, or scenarios that effect the system. The resultant consequences and behavior of the system manipulation are analyzed and further changes are considered and acted upon. Through these iterations of experiments, experience is developed and a final decision or policy is settled upon.

This method, often called "trial and error," has its drawbacks. Trial and error on the actual system can be expensive, time consuming, and even detrimental. Thus, one can rarely perform experiments in the business world. Even if the opportunity exists, if the system does not yet exist, experimentation is not a feasible alternative. A manager must apply other methods to develop experience.

Analytical techniques are another method for decision analysis and involve the application of mathematical equations that have been derived by management scientists. The system in question is studied and a mathematical model is constructed that represents the interactions of the system and environment. The relevant equations are solved using simultaneous equations and calculus techniques resulting in an optimal solution. An optimal solution is the best solution among several feasible solutions. The manager can then institute his/her decision or policy. However, a few caveats must be considered when a manager chooses the analytical approach. First, the system may be amenable to a mathematical model but deriving the solution may be beyond the capabilities of the manager or his/her staff. Second, as a model more closely simulates reality, i.e., becomes increasingly more complex and mathematical techniques becomes incapable of fully describing the system. Thus, another tool is required to resolve these dilemmas.

B. SIMULATION FOR RESOURCE ALLOCATION

To solve analytical enigmas, the next alternative available to managers is the world of simulation. Simulation methodology is the development of a mathematical model or a series of models that describes the behavior of a system over time. Thus, simulation is heavily dependent upon analytical techniques and offers a method of solving analytical problems that are beyond the manager's capabilities. Many other benefits exist with simulation, but simulation is not an end in itself. It is a vehicle from which data for further analysis is collected and conclusion drawn. It does not replace the experience and intuition of the manager but instead it is an augmentation to the information available to a manager.

The two preeminent advantages of simulation in comparison to previously discussed methods are imitation of reality and reduced expense. Imitation of reality is a key concept and an advantage of simulation because it allows a manager to observe the behavior of a system as he/she induces change without agitating the real system. Thus, the difficulties encountered with actual experimentation are mitigated. The manager can now observe behavior of a simulated system and will be able to determine the system's sensitivity to changes in key variables, locate critical factors or problem areas, and evaluate the effectiveness of his/her decisions. Thus, a manager can derive effective solution before the actual implementation of an unproved policy or action. Also, during a simulation exercise, he/she can control many features of a system that he/she would not usually control in a real world setting. A manager can develop experience by relating known manipulation to known results.

Concerning expense, simulation offers several economical benefits. By use of digital computer simulation methods, a manager can evaluate alternative ways of meeting objective in a fraction of the time that would normally be required for the long term effects of a proposed decision to occur in time. Time is money and simulation methods allow the manager to be in control of time. He/she can compress time so as not to wait for the passage of time to produce results. Second, manipulation does not occur with an actual system in which costly mistakes and pitfalls could occur. He/she can gain experience at the expense of a simulation model vice the organization's capital. Furthermore, if the system is in the design stage and does not yet exist, simulation allows for cost benefit analysis of hypothetical situations. The manger can derive an economical design founded upon simulated results. Construction of a system need not be based upon an intuitive guess or a faulty experience. Also, simulation produces data inexpensively which can be used for further analysis.

Therefore, simulation is a viable and essential tool for approaching all but the most simplistic of resource allocation problems. Simulation methodology provides the manager with the best of all feasible alternatives without committing an organization's capital. Additional consideration must be given to the low cost associated with simulation. The next section discusses how a manager applies simulation as a tool.

C. SIMULATION THROUGH COMPUTERS

Assuming the manger understands the benefits of simulation methodology, how does he/she most effectively implement simulation techniques? Simulation can range from very simplistic methods involving the development of a solution by hand to the most complex that require the application of digital computers and simulation specific languages, i.e., GPSS and SIMSCRIPT to name a couple. Unfortunately, for a majority of managers, the hand technique becomes either too complex and time consuming or the computer oriented method is too expensive and beyond the manager's skills. Thus, he/she must consult a simulation programming expert to solve his/her problems, if one is available. Another approach would be the marriage of simple and complex techniques through the application of off-the-shelf computer spreadsheets to solve all but the most difficult simulation quandaries.

Digital computer spreadsheets were first introduced with the development of *Visicalc* by Dan Bricklen and Bob Franston in 1979 [Ref. 1]. *Visicalc* did little more than replace pencil and paper calculations with a computer. However, during the past decade, computer oriented spreadsheets have grown exponentially in functionality and computing power. Through spreadsheets, a manager has the ability to produce powerful simulation models. All the manager needs is knowledge of simple spreadsheet procedures and an understanding of basic principles of simulation methodology. Currently, the most popular off-the-shelf spreadsheet software is *Lotus 1-2-3*, followed by *Excel*. Each provides limited simulation ability by employing "what if" analysis. Thus, simple simulation is available to anyone who owns a microcomputer and a spreadsheet package.

However, "what if" analysis is limited to a few variables and does not address time or probabilistic issues that are essential criteria for simulation methods. These limitations can be resolved through the purchase of inexpensive simulation "add-in" programs such as *Simulated Solution*, *@RISK*, or *Crystal Ball*, or by spreadsheet programming through "macro" commands. Add-in programs are not widely available to most managers, but the basic spreadsheet program is available to virtually all managers. Accordingly, Chapter III will explore the applicability and development of computer spreadsheets for solving moderately complex simulation problems using simulation methodology and spreadsheet macros. In addition, guidelines for the development of computer spreadsheet simulation models will be identified for application to other resource allocation problems.

III. SIMULATION METHODOLOGY

This chapter outlines the methodology entailed when developing and applying simulation models and digital computer spreadsheets to resolve resource allocation predicaments. The chapter will begin by summarizing relevant simulation oriented terminology followed by a discussion of an effective strategy for a manager who embarks upon the creation of a simulation model. When appropriate, concepts, suggestions, and command specific to *Excel 4.0* (adaptable to other off-the-shelf spreadsheets) will be provided to assist a manager on his/her macro programming endeavors.

A. TERMINOLOGY

The following terms are employed throughout the course of this study and will be defined here so as to avoid any confusion in terminology.

1. *System:* The system is any set of interdependent elements that function within an organization to meet specific goals. A system may have subsystems.

2. *Model:* The model is an imitation of a system using formulas, logic statements, etc., that when conglomerated represent how the system physically interacts within reality.

3. *Discrete:* A discrete system has events that occur during a specific point in time. Time is considered as a distinct unit vice continuous with events flowing from one to the next.

4. *Stochastic:* A stochastic system entails estimates on the part of the decision maker of events or variables that are random or probabilistic in nature.

5. *Deterministic:* When variables are assigned a single-valued estimate vice a stochastic estimate, they are considered deterministic.

6. *Exogenous Variable:* Exogenous variables are entered into a model and are not altered in value during the simulation exercise. An exogenous variable is also referred to as an *environmental variable*.

7. *Endogenous Variable:* Endogenous variables that are dependent upon the interactions within the simulation model. Their values are derived by the model during the simulation and are often referred to as *state variables*.

8. *Policy Variable:* Policy variables are variables that obtain their value as a direct result of the decision makers' intervention.

9. *Flow Chart:* A flow chart is a graphical representation using boxes and arrows to represent events within a system as events progress through time. Interactions between variables, environment, etc., are captured within a flow chart.

10. *Feedback:* Feedback is the transferring of output back to the input so that policy variables can be altered in an attempt to obtain a desired output.

11. *Routine:* A collection of computer commands that perform a function or functions.

B. SIMULATION STRATEGY

Simulation philosophy is a methodology of approaching management allocation predicaments. Unfortunately, there is no specific procedure that a manager can apply due to simulation models being unique in application and must be designed anew with each new kind of problem. However, there are commonalties between problems and models that facilitate a strategy when a manager confronts the task of simulation. Thus, during the construction of an effective simulation model, the person who develops the model will proceed through a logical progression of steps. Some steps will be easier than others and consume less time while others are more difficult and time consuming. However, all are necessary as each step depends upon the preceding one. The remainder of the chapter will lay out the steps required for an effective strategy when addressing all but the most complex simulation scenarios.

1. Define the Problem

Essential to the success of a simulation model is the definition of the problem that the manager wishes to simulate. He/she must pose the question: "What is my objective for the model?". The objectives must be clearly stated so that the manager or programmer can assess the purpose of the model with its resultant desired output. Initially, the definition may be broad, such as a plan to minimize cost with the system. As the model progresses through the following steps, the objectives will become more narrowly defined as different aspects and objectives are realized. However, with a defined objective, a programmer can proceed to the next step of charting the interactions within the system.

2. System Flow Charts

Capturing the essence of a system that is required to properly develop a simulation model is best done through flow charts. Before describing flow charting techniques, one must first understand basic system relationships. Figure 3.1 illustrates the

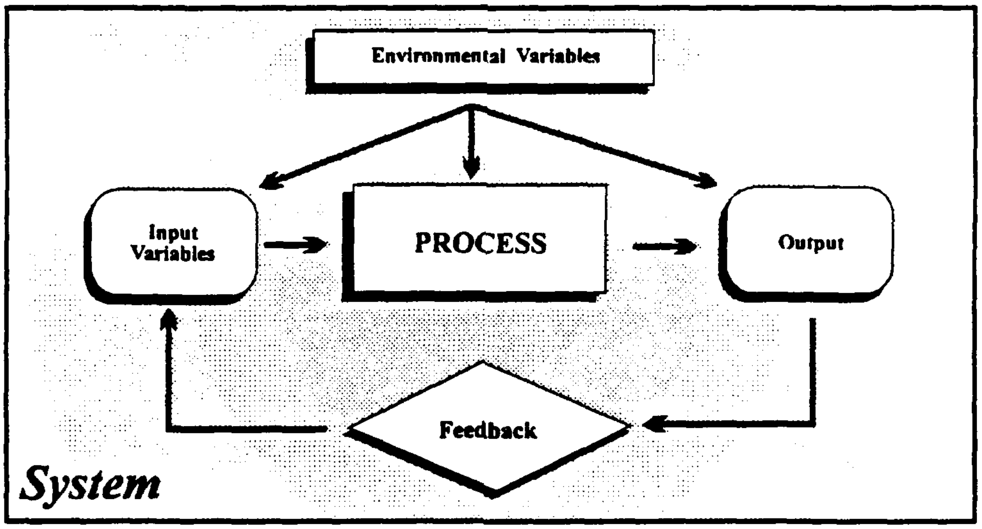


Figure 3.1: System Interactions

interactions that exist within a system. Key to the success of the simulation model is the proper identification of interactions shown in Figure 3.1. The *environment* is beyond the control of the decision maker but it interacts with all facets of the system and must be understood. The decision maker controls the *inputs*, monitors *feedback*, and makes decisions through policy variables. The *process* is the heart of the simulation model where most of the functional interrelationships are understood and then programmed. With a stated objective and a basic understanding of the system in terms similar to Figure 3.1, a basic flow chart can be constructed. [Ref. 2:pp. 2-4]

The first flow chart will capture the essence and general interactions that are involved in the system. It should be a relatively simple chart as it will become the foundation from which all other interactions and flow charts are constructed. The first flow chart will be the backbone of the simulation model and it is upon this that a master control routine is created. Subroutines branch out from the master routine to perform more specialized tasks or functions. This logic of breaking the system into one master routine and several subroutines is ideal for "debugging" a model. The technique of routines and subroutines is discussed later.

When developing flow charts, a manager should only consider key variables and interactions. Once a working model is created, more complexity and realism can be added as needed. The greater the number of endogenous, stochastic, and policy variables included in a model, the more complex the model becomes. This results in an increase in the time required to develop and run a simulation model. However, with an increase in complexity the model becomes more accurate. The more accurately that the model represents reality, the more accurate the results and the closer the simulation will be to meeting the objectives. This trade-off between time and accuracy is partially determined by the objectives and partially from the experience of the programmer. With the flow charts drawn and thoroughly understood, the programmer continues to the next step of simulation strategy.

3. Mathematical Modeling

Mathematical modeling often coincides with the creation of flow charts. This occurs because mathematical models involve functional relationships where exogenous variables (inputs) are transformed into values for endogenous variables (outputs). Also, the functional relationships will be an integral part of the boxes within flow charts. Mathematical modeling is also the point in the simulation process where the discipline of spreadsheet modeling becomes a consideration. A programmer must be continuously thinking about how he/she is going to program the spreadsheet to recreate the mathematical and functional relationships.

During the mathematical modeling phase, a number of sources are referred to for equations and relationships. Several equations will be used that have been created by management scientists for analyzing a problem using analytical techniques. This is the case for the inventory distribution and queuing models discussed in the next two chapters. Other sources will be from the discipline from which the model is formulated such as general accounting relationships. Other functional relationships are developed by understanding the relationships inherent within the flow charts created for the system. The purpose of mathematical and functional relationship modeling is to describe the system as carefully as possible. Each equation or relationship describes a relationship between two or more factors of interest in the system. When consolidated, they represent the flow charts and eventually the complete system.

Critical to the mathematical modeling phase are the assumptions that are built into the simulation model. What is considered to be generally understood and in what situation will the simulation model be exercised? A caveat must be considered at this point in the discussion. Anyone who deals with simulation must understand that the model is only as genuine as the input and the assumptions inherent to the input. A poor assumption will invalidate the model no matter how well the model is constructed. As the saying goes, "Garbage in gives garbage out!" Therefore, the importance of this step in model development cannot be over emphasized.

Another consideration that must be understood during simulation construction is the unexpected. A model should take into account every conceivable value of the system being considered. For simplistic models, it is easier to prevent values or place limits within the model. For example, if the stock level in an inventory system achieves a certain level, the simulation should perform a function or end simulation. The reasoning behind this logic is the difficulties a programmer will encounter if a condition is forgotten and is encountered during the simulation. Numerous hours of debugging can result from an unaccounted value or a condition that results in misleading and invalid output.

4. Creating a Spreadsheet Simulation Macro

This step is the point of divergence from a typical approach to simulation methodology. Instead of applying a spreadsheet to solve a simulation oriented problem, the traditional approach is for a manager to choose hand simulation or simulation specific computer languages to solve his/her resource allocation dilemma. If the problem under consideration is a continuous situation vice discrete, simulation specific languages are the optimal choice. However, the focus of this thesis is on moderately complex, discrete scenarios that lend themselves to spreadsheet simulation. The remainder of this chapter will be oriented towards the application of a spreadsheet for resolving of resource allocation problems. In particular, techniques developed during this study will be presented to the reader to augment his/her spreadsheet simulation endeavors.

However, before launching into the programming phase, the programmer must become intimately familiar with the capabilities of the spreadsheet he/she intends to employ. Spreadsheets are not designed to be used for simulation tasks but are fully capable to do so through macro programming. However, one must be creative in programming the spreadsheet to perform simulation. The more that a programmer understands the capabilities of his/her spreadsheet, the easier the task will be.

After constructing flow charts and mathematical relationships, the next task, often the most time consuming, is the creation of the simulation macro(s) and worksheet within a spreadsheet program such as *Excel 4.0*. Careful preparation and forethought will save the programmer several hours of debugging during the creation of the program. However, debugging will be required no matter how efficient the programmer is. The following paragraphs are techniques that will help the manager solve resource allocation problems using *Excel* or other *off-the-shelf digital computer spreadsheets*.

a. Subroutines

To facilitate debugging, ease of understanding, and use of flow charts, a "master" macro should be created. The master macro will control the entire simulation process through a network of supportive subroutines. The master routine should be constructed upon the basic or central flow chart. It should control time and its respective iterations along with output and input of variables. The subroutines should be developed to perform specific or several functions of the flow charts or separate blocks within flow charts. The concept of "block-building" through subroutines facilitates debugging by limiting the areas where a programmer must look for difficulties.

A subroutine does not have to be part of the same macro. It can be its own separate macro that is initiated by the master or other subroutine macros. Separate macros become a necessity with complex scenarios that require many lines of programming code. Otherwise, the program becomes too large, difficult to debug, and some functions of *Excel* become limited or lost. Also, *Excel's* group editing abilities allows the editing of several macros simultaneously, thus saving time.

b. Variables

Several techniques exist for assigning values to variables. For exogenous variables that are deterministic and not changed for different scenarios, the best command to use is **SET.VALUE(*reference, values*)**. For variables that need to be updated during simulation and recorded in a separate location on the spreadsheet, the best command to use is **FORMULA(*formula_text, reference)***. Exogenous and deterministic values that are changed by the user for different scenarios or assumptions are entered through a function referred to as **DIALOG.BOX(*dialog_ref*)**. This command presents the simulation user with an interaction box similar to Illustration 3.1. Dialog boxes can be used for a number of other functions such as prompting the user for input needed for policy variables.

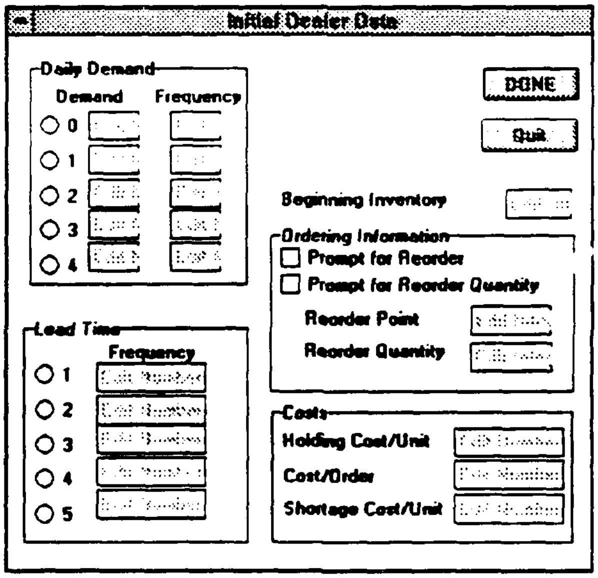


Illustration 3.1: Sample DIALOG BOX

Another function of *Excel* that is used for variable manipulation is the **IF(*logical_test, value_if_true, value_if_false*)** command. Using this command, variables (i.e., cell reference to itself) can cumulate data when a condition is either true or false as shown in Illustration 3.2. **IF(..)** is also used for stochastic events and Monte Carlo techniques. By using multiple **IF(..)**s, a stochastic variable that has been converted to a cumulative relative frequency is coupled with a **RAND()** function to generate a random number less that one. The **MAX(*number1, number2,..*)** function then determines the value. This process is depicted in Illustration 3.3.

	A	B	C
	name	***command***	***comments***
333	Total_Type_1	=IF(Patient_Type=1,Total_Type_1+1,Total_Type_1)	Tally Patients
334	Total_Type_2	=IF(Patient_Type=2,Total_Type_2+1,Total_Type_2)	
335	Total_Type_3	=IF(Patient_Type=3,Total_Type_3+1,Total_Type_3)	

Illustration 3.2: Data Culmination

	A	B	C
	name	***commands***	***comments***
67		=RAND()	Determine Daily Demand
68		=IF(B$67>(VLOOKUP(0,Demand_Table,4)),Demand_1,Demand_0)	
69		=IF(B$67>(VLOOKUP(1,Demand_Table,4)),Demand_2,Demand_0)	
70		=IF(B$67>(VLOOKUP(2,Demand_Table,4)),Demand_3,Demand_0)	
71		=IF(B$67>(VLOOKUP(3,Demand_Table,4)),Demand_4,Demand_0)	
72	DEMAND	=MAX(B68:B71)	

Illustration 3.3: Stochastic Event

c. *Naming Variables and Locations*

A powerful capability of spreadsheets is the ability to name cells or blocks of cells. Thus, a cell that is being used as a variable can be assigned a name.

Naming should be used as much as possible to permit the programmer, debugger, and user to better understand the program. Instead of trying to determine which cell a formula is referenced to, a name is utilized. DEMAND and Demand_Table are examples of using names as shown in Illustration 3.3.

d. Recording Data

Computer simulation involves the passage of simulated time in a compressed format. The gathering of output statistics during simulation requires an ability to record data either in computer memory or on the spreadsheet for further analysis. This can be done through a couple of methods. The first is to collect data using **IF(..)**, **MAX(..)**, etc., functions as shown in Illustration 3.3. The disadvantage of this method is that the data for each particular iteration is lost. However, it is ideal for simplistic Monte Carlo methods or for the summation of information.

The second method is to record the information in a spreadsheet matrix format. Unlike simulation languages that support three-dimensional storage of information in computer memory, a spreadsheet requires the recording of information after each iteration onto the spreadsheet. This is the major difficulty that was encountered when applying spreadsheets for simulation problems. It was not impossible but requires some creative programming to resolve some of the perplexities encountered.

When using a matrix format to store data, one needs to have the ability to reference a location on the spreadsheet. The **OFFSET(*reference, rows, cols, height, width*)** command was used extensively for this function. For the reference, the corner cell of the matrix was assigned a name for easy reference. Therefore, a two dimensional matrix was accessible on the worksheet for data storage and manipulation.

e. Manipulation of Time

The method by which time is controlled within the macro that controls the spreadsheet simulation determines the foundation upon which all macros are constructed. The idea is to move the model through time to see the dynamic behavior of the system. The simulation begins at time zero where all parameters have their initial values as provided by the user. As time progresses during simulation, various events occur causing changes within the model. Thus, time is central to the simulation. Two methods of time management were used in the examples in the following chapters.

The first method is to allot time into fixed units. Time is then iterated using the **FOR(*counter_text, start_num, end_num, step_num*)** function with its corresponding **NEXT()** until the user inputted time limit is reached. This was used for the simulation of inventory distribution and financial management problems.

The second method is a "next event" technique [Ref. 3]. Time is not iterated in fixed units but instead is iterated by the time required until the occurrence of the next event. Next event methods require more creativity in program design so that each event can be traced with its respective information. The method employed in the queuing examples was a two-dimensional pointer-matrix methodology Each event was assigned a time and a pointer that maintained the location of relevant data.

f. Logic Statements

For situations that require a separate set of actions based upon different conditions, *Excel* offers several logic functions. For example **IF(*logical_test*)**, **ELSE.IF(*logical_test*)**, and **ELSE()** allows the program to execute separate functions due to three separate conditions. The function **WHILE(*logical_test*)** with its corresponding **NEXT()** permit localized iteration routines if required. Each of these functions lends themselves to conditional requirements within flow charts.

These above techniques are the major methods that were developed during the creation of the models contained in the next three chapters. Other techniques can be discovered in the program listing contained in Appendices A, B, and C.

5. Validation of the Model

Once a working spreadsheet model is created, the most difficult task for the programmer is validation and debugging of the model. Essentially, given inputs with corresponding known correct outputs (this can come from an organization's past data or hand calculations) are entered into the model and the model's output is scrutinized. Other methods of debugging also can be employed. *Excel* has a built-in add-in function for debugging. This add-in permits the insertion of breaks within the macro or check points where values of critical variables can be called up and their validity determined. Another function that was found to be invaluable was the macro utility bar. Contained on this utility bar is a function that allows an individual to "step" through each individual line of the macro. Thus, a programmer can validate that every line of programming in the macro is performing what it was designed to do. The utility bar also allows the pausing of a macro. This facilitates repositioning of the computer display of the macro or worksheet to check on other sections of the spreadsheet during simulation execution.

One who begins programming will learn that the validation phase is the most difficult and rewarding. As he/she progresses through every conceivable scenario and situation, a simulation model will be created. Often, at this point, further complexity is added to the working model to enhance the accuracy of the model until the final simulation model is created.

6. Model Implementation

Once a model is created and validated, the model should be put to use and the benefits of the simulation realized. Often the model is further modified to add more realism or to meet new objectives. One will find that the first model is the most difficult. Fortunately, as time progresses and experience is gained, the process becomes much easier.

The next three chapters present the application of simulation and computer spreadsheets for analyzing resource allocation problems. Techniques discussed in these chapters are used to create spreadsheet simulation models for three unique allocation scenarios frequently confronted by managers. Each will demonstrate how "what if" analysis through spreadsheet simulation can provide economical guidance to managers when making crucial policy decisions. Spreadsheet simulation provides management with the capability to analyze how a decision effects a dynamic system without the expense of trial and error.

The simulation systems chosen for illustration and discussion are *inventory distribution management, queuing environment utilization, and a financial budgeting* scenario. Before discussing each simulation model, an introduction is provided outlining the dilemmas faced by managers and the apropos solutions provided by management science theory concerning the aforementioned illustrations. With an understanding of the traditional solution to each resource allocation problem, the alternative approach of simulation will be introduced to illustrate how simulation can augment management science theory. The logic behind each spreadsheet simulation model will be discussed followed by scenarios demonstrating the power of each model. Each scenario's result will be analyzed to conclude each section.

IV. INVENTORY DISTRIBUTION MANAGEMENT

This chapter is the first of three chapters that present applications of simulation and computer spreadsheets for analyzing resource allocation problems. The first resource allocation to be discussed is a dilemma that often confronts managers. How to effectively develop a set of rules and policies for managing inventory? Inventory can be best summarized as the items that are maintained in storage to meet the immediate and future demands within the organization or by customers. Almost all institutions possess some form of inventory. Inventory can become extremely large and consume a majority of the capital assets available to the manager. Therefore, the expertise of maintaining inventory capital at a minimum while simultaneously ensuring that the demands of the organization and its customers are satisfied is a crucial facet of a successful manager.

A. THEORY

Inventory management is an aspect of management science that has been analyzed and documented in literature for decades. Management scientists have developed several analytical models to assist a manager in achieving a balanced and economical inventory management system. The central thrust of these models is to apply analytical techniques to achieve a theoretical balance between desirably low inventory levels with that of sufficient stock to meet customer requirements. Ideally, the analytical results will allow a sagacious manager to achieve positive customer relations and the lowest possible commitment of assets. This requires few or no backorders with low inventory levels. Unfortunately, analytical methods are formulated under the presumption of ideal conditions such as predictable demand and lead time. With the introduction of reality, the manager must augment an ideal solution with experience or other methods to obtain an efficient inventory system. It will be shown that spreadsheet simulation can be a valuable tool in enhancing a manager's experience.

Before introducing inventory simulation, a manager must understand inventory management theory. The overall objective of the analytical method is to achieve the lowest commitment of assets. The commitments of capital in the form of costs are as follows:

1. *Holding* or *carrying costs* that are essentially the expenses of physically having an inventory within the organization. Examples of these costs are floor space, insurance, and obsolescence. Holding costs do not include the actual value of the inventory. [Ref. 2:p. 56]
2. *Ordering* (or *setup* in a production environment) *costs* that are incurred with each decision to order (or produce) more inventory. Examples of these costs are clerical costs associated with processing an order, shipping cost, and material handing costs once the order is received, or the costs of restarting the production line after a temporary shut-down. [Ref. 2:p. 57]
3. *Stockout* cost that consists of forgone profit, lost sales, or the cost of an emergency order associated with the inability to meet customer demand when an item is not available. This cost is the most difficult to estimate and often is the most expensive. [Ref. 2:p. 57]

Once a manager has assessed the costs involved in maintaining an inventory, an understanding of how the costs interact is required to illustrate the purpose of analytical techniques. The first interaction is that holding and order costs move in opposite directions. Assuming ordering costs are greater than holding costs, a manager who makes large orders will decrease the total cost of ordering. However, large orders result in higher inventory level, thus increasing holding costs to the point that will eventually exceed the benefit of reduced ordering costs. The second interaction is that stockout costs will decrease as order size increases but are more a function of the reorder point vice quantity. If a manager maintains his/her inventory levels low to reduce holding costs, the potential exists to order too late, resulting in items being out of stock with a corresponding exponential increase in stockout costs. These conflicting interactions lead the mangers to ask themselves two questions: How much should I order? and When should I order?

The traditional method in management science for resolving the manager's questions is to apply analytical techniques referred to as the Economic Order Quantity (EOQ) model. This model, with algebra and differential calculus, will identify the most economical balance between order quantity and reorder point. However, as previously discussed, adding more realism and complexity to a scenario will result in the traditional analytical methods of analysis becoming prohibitively difficult to apply, even for professional mathematicians and statisticians. Thus, to cope with reality, managers rely upon experience to determine a sufficient safety or buffer stock. However, a buffer stock result in a higher reorder point and excess inventory. Determining the level of safety is not an easy task because the best set of rules often cannot be established in advance. The rules for determining an appropriate safety stock must often be arrived at through the process of trial and error. However, by applying simulation and spreadsheet analysis, a significant reduction in the costly methods of guess work and trial and error can be achieved by the manager.

B. INVENTORY DISTRIBUTION SIMULATION

The inventory distribution system model that was developed using Excel 4.0 is designed to simulate a typical factory to dealer distribution system with multiple levels of warehouses. A discussion of the logic behind the model is included in this section. A full listing of the model's macros and comments are provided in Appendix A. The model is designed for analysis of multiple scenarios. A relatively simplistic model consisting of one dealer with non-probabilistic lead and demand times demonstrating EOQ theory can be analyzed. Additionally, the model is capable of allowing the user to analyze extremely complex scenarios with up to three warehouses, probabilistic demand and lead times, and continuous involvement by the user. Complex scenarios illustrate the difficulty of applying EOQ techniques for achieving optimal solutions. Many different aspects were considered and included within the spreadsheet model. Furtl nodifications to the macros can be performed to include any degree of realism within the simulation. However, the extent of modifications to the model for further capturing the richness of a real-world situation should not be so complex that the user cannot understand or appreciate the spreadsheet simulation model for analyzing a managerial problem.

Before constructing the inventory distribution simulation, one must first determine the objectives of the model. What do we wish to examine? The following model was designed to analyze inventory level, backorder and cost behavior that are the critical aspects of inventory management system. A model that demonstrates these behaviors will augment a manager's policy concerning when and how much to order under different scenarios.

With the model's objectives in mind, a number of flow charts were created to build and chart the sequence of events that were required to properly understand an inventory distribution system. The first flow chart developed gives a broad summary of how orders generated by the customer are received by the dealer and then progress down through the distribution system to the factory. The result is merchandise being provided to the customer. This progression of orders from the customer to the factory is illustrated in Figure 4.1. The flow of events is different if less than three warehouses are included in the simulation. Instead of the orders and inventory flowing between warehouse #3 and the factory, this event may occur between warehouse #1, #2 or more, depending upon the number of warehouses simulated.

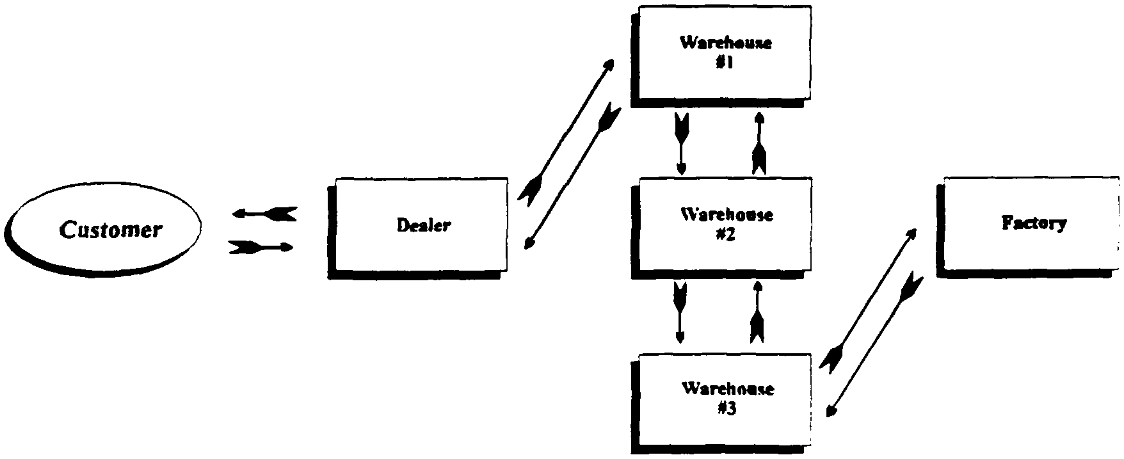


Figure 4.1: Inventory Flowchart

The flow of events is a generic representation of inventory distribution systems which can be found in the private sector or military logistic supply systems. For example, the supply department on a ship performs the same function as a dealer in the private sector. The shipboard supply department receives customer demand from the onboard customers it was designed to serve, such as O-level maintenance facilities.

Key to the flow of events is the treatment of "time" within the model. For an inventory system, time is a fixed unit. Thus, each iteration through Figure 4.1 is the passage of one time unit and all other elements are treated as variables. On the other hand, a queuing model, which follows this chapter, treats time as a variable with other elements being fixed. Thus, time becomes the critical element in the development [illegible] simulation model and will lead to vast differences between models.

To simulate an inventory distribution system with time as a fixed unit, one worksheet and six macros were created within Excel 4.0. The worksheet is a place holder for the data and graphs created by the individual macros. The worksheet accesses the macro by two macro command buttons. One button begins the simulation routine, while the second transfers the data to the graphs for analysis. The six macros perform the actual computational work of the simulation. Five of these six macros are for each of the inventory management centers depicted in Figure 4.1 except that the customer is included within the dealer macro. The sixth macro is the master macro that controls the process of interactions among the five individual inventory management macros and is the first macro to be executed when the user begins the simulation from the worksheet

As shown in Figure 4.2, the first action by the master macro is to display an initial input dialog box requesting information on inventory status and policies such as reorder point, order quanity, holding costs, etc. This is the information required for performing a simulation. Additionally, the number of warehouses desired and how much user involvement is needed during the simulation is requested by the input dialog box. Options

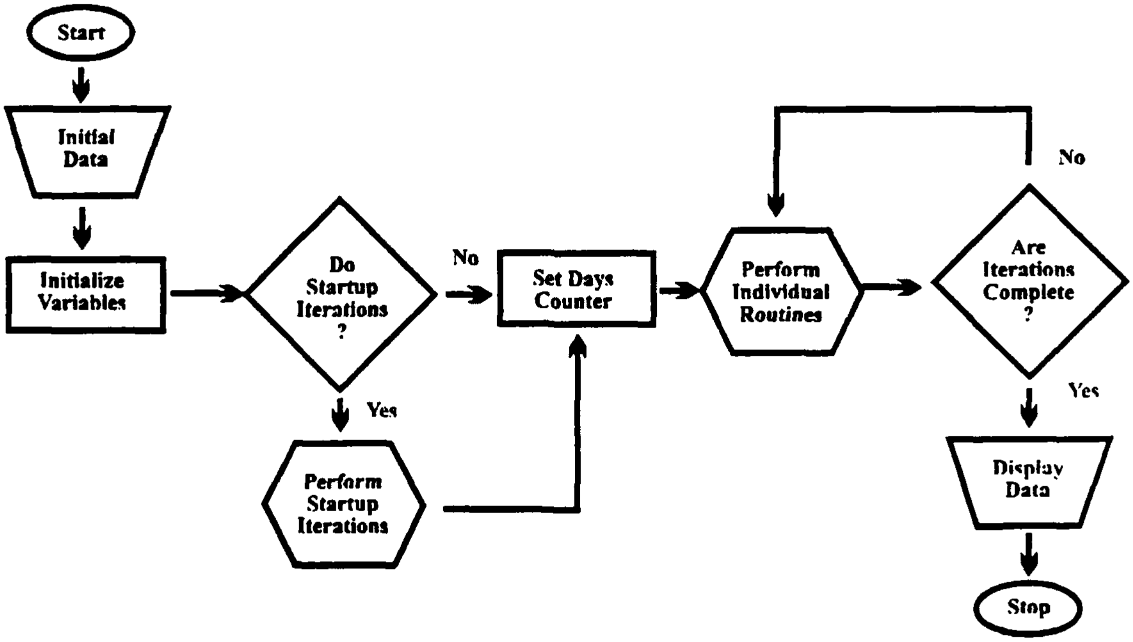


Figure 4.2: Master Macro Flow Chart

of involvement range from none, with the exception of initial data, to complete control requiring every decision to be made by the user during the simulation. Once the master macro has the information required to perform the simulation, it begins an iterative process of progressing through each macro as shown in Figure 4.2.

If startup iterations are desired, the master macro will iterate without any user involvement thus producing random inventory and backorder levels. This option is useful for the full involvement scenario as it adds another level of complexity to the problem. Once the initial data and startup segments are performed, the macro begins an iterative process for the number of days requested by the user. The master macro's first step in the iteration is to call upon the dealer macro that initiates the inventory distribution process. When the dealer macro returns control to the master macro, an iteration is complete and time is incremented by one unit. The process then repeats and continues until the number of iteration days is complete. The last function of the master macro is the transferring of data back to the worksheet.

The dealer macro initiates the inventory distribution system. It, along with each of the warehouses, has virtually the same flow chart of events occurring as illustrated in Figure 4.3. The first calculation that is performed by the dealer but not the warehouse macros is to calculate customer demand that is either probabilistic or deterministic. With demand known, the macro will then fill any backorders if inventory is present. What is left of current inventory is applied to the customer's order and if insufficient, a backorder log is developed. With the remaining inventory, the macro makes the determination if inventory is below the reorder point. If this is the case, an order is placed at the warehouse. The order will be received by the first warehouse depending upon the lead time that is probabilistic or deterministic. Even if there is no order placed, the dealer macro calls upon the warehouse macro at this point to permit updates of the warehouse and factory inventory levels. The last function of the macro is to determine if a previous order has arrived. If so, it is the added to the inventory.

The warehouse macro differs from the dealer macro in that it does not calculate customer demand but instead receives its orders from the dealer. Furthermore, each warehouse will call upon the next warehouse in the distribution chain or the factory, depending upon the number of warehouses in the simulation.

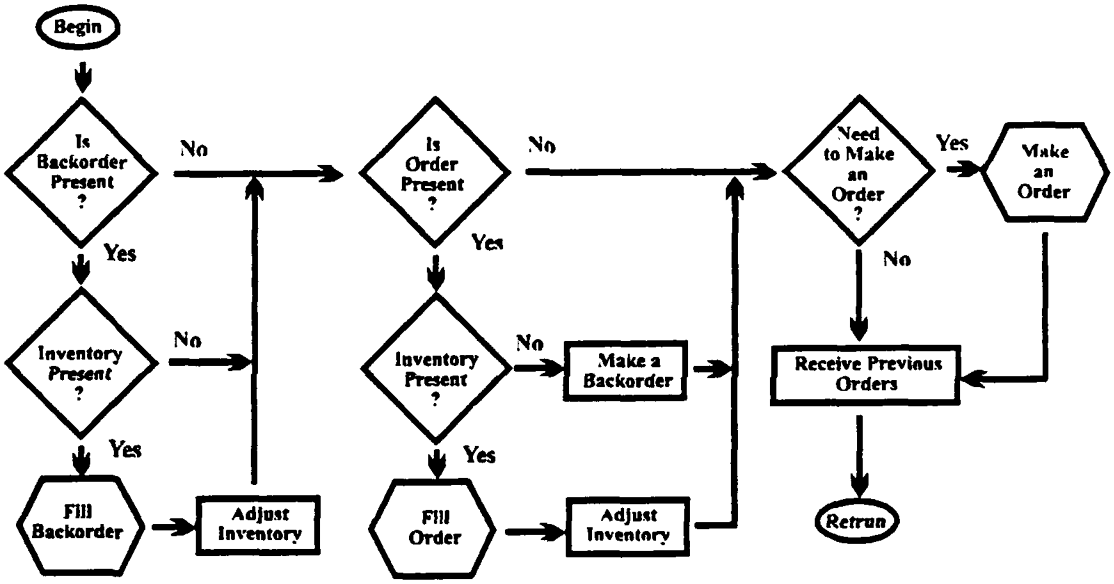


Figure 4.3: Dealer and Warehouse Macro Flow Chart

The flow of events for the factory macro is illustrated in Figure 4.4 and is slightly different in design and concept than the dealer and warehouse macros. Instead of making the decision to place an order, the factory macro makes the decision to either startup or shutdown production. The same logic as reorder point is applied. Production will begin when inventory level depletes below a certain level and production stops when inventory exceeds a certain level.

Common to the entire macro chain is the concept of macro nesting. This means that customer's order precedes through the inventory management system in a linked like manner. The order starts at the dealer on Day 1 and continues through the warehouses reaching the factory at an unpredictable date. This time-delay effect on each inventory management point amplifies the uncertainty in customer demand and illustrates the need to use simulation models to examine the effect of different inventory policies at different management points. Once at the factory, the order reverses direction back to the dealer as illustrated in Figure 4.1. Another element of the system is that each successive iteration within the model is the passage of one time unit that equates to a single day. However, the user can assume any length of period as long as input data regarding demand rates, lead time, cost parameters, and so forth are appropriately scaled. The output will then reflect the time unit chosen by the user.

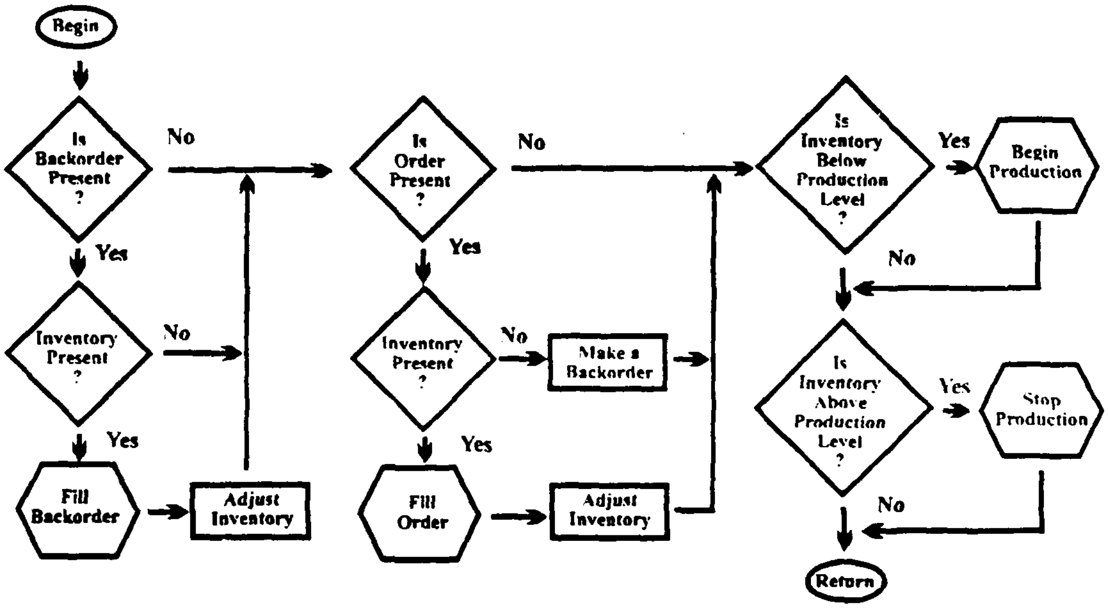


Figure 4.4: Factory Macro Flow Chart

C. SCENARIOS

With an understanding of the logical progression of events and how functions of each macro, four different scenarios will be analyzed to demonstrate how the simulation spreadsheet model can be used to augment the manager's expertise concerning inventory management. Each scenario could be solved using analytical and hand simulation techniques. However, as the level of complexity increases, it becomes apparent how a computer spreadsheet facilitates multiple "what if" analysis in a fraction of thc time required to obtain one analytical answer. Each simulation will be a dynamic system showing the interactions and oscillatory behavior typical of a dealer to factory inventory distribution system over a sixty day period. Sixty days with initially low levels of inventory are chosen to force an oscillatory behavior to occur early within the simulation. The purpose of each scenario is to demonstrate how the spreadsheet simulation technique can be applied to illustrate oscillatory behavior and derive possible optimal solutions vice gathering reams of data.

The oscillatory behavior may be transparent to a manager involved with only one aspect of the distribution system. He/she may not appreciate how small changes in retail demand often create large swings in factory production and warehouse inventory that far exceeds the fluctuation of retail demand. These fluctuations can be quite costly due to employment instability, over capacity, and high inventory levels. Through simulation, a manager can manipulate variables without experimenting on the actual inventory levels He/she will see how simple changes in variables and policies can have dramatic effects upon the system as a whole. The three scenarios chosen manipulate variables only slightly, yet each has its corresponding oscillation.

The first three scenarios will begin with the same initial data but will have varying levels of complexity. The first two will involve no user involvement with the exception of entering initial data. The first scenario will have only one warehouse between the dealer and the factory while the second will have three warehouses. The third scenario will also have three warehouses but will demonstrate the maximum complexity capable of the simulation model. There will be ten startup iterations to allow for random levels of inventory and backorders. Additionally, all orders and factory production decisions will be decided upon by the user during each iteration. The fourth scenario will be similar to the second but instead lead times will be significantly reduced. This reduction in lead times will demonstrate a Just-In-Time (JIT) inventory distribution system. The results of each simulation scenario are discussed in each section and complete output graphs are provided in Appendix A.

1. One Warehouse, No User Involvement

For each of the four scenarios, the same initial input data was used as shown in Table 4.1. The initial reorder points and quantities are chosen to be large and late so that an initial oscillatory behavior is demonstrated early within the model. Furthermore, the factory's inventory level is twice that of the others to absorb the initial surge in demand. The factory's production rate is sufficient to exceed demand, therefore providing an illustration of startup and shutdown events at the factory.

TABLE 4.1: INITIAL INPUT DATA

	Dealer	Warehouses	Factory
Beginning Inventory	50	50	100
Reorder Level	25	25	*
Begin Production Level	*	*	35
Stop Production Level	*	*	100
Amount of Order	25	25	25
Rate of Production	*	*	30
Holding Cost	$.10	$.10	$.10
Order Cost	$20	$20	*
Setup Cost	*	*	$100
Shortage Cost	$50	$50	$50

Having only one warehouse between the dealer and the factory is the simplest of the four scenarios, thus one would expect very little oscillations. This would be the case if demand and lead time were not probabilistic. A simple EOQ calculation would allow the dealer and the managers of the warehouse to determine economic order point and level resulting in relatively stable inventory patterns. However, all four scenarios have both demand and lead time being probabilistic as provided in Table 4.2. The latter column of relative frequency lead times was used in the final scenario to represent a JIT environment.

Having probabilistic demand and lead times without user input results in multiple orders being placed before receiving any shipments from previous orders. This is the primary reason that large oscillations in inventory levels occur as illustrated in Figure 4.5.

TABLE 4.2: DEMAND AND LEAD TIME FREQUENCY

Demand	Relative Frequency	Lead Time	Relative Frequency
4	20	1	20/90
5	20	2	20/10
6	20	3	20/0
7	20	4	20/0
8	20	5	20/0

Additionally, even after achieving large inventories by day twenty, strong demand with corresponding late ordering again results in a depletion of the dealer and warehouse inventory by day thirty. Consequently, the dealer and especially the warehouse develops significant and expensive backorders. By altering the reorder point and levels, an optimal solution can be obtained. However, by adding levels of warehouses to the scenario, the difficulty in predicting demand and oscillatory behavior make obtaining a solution even more illusive.

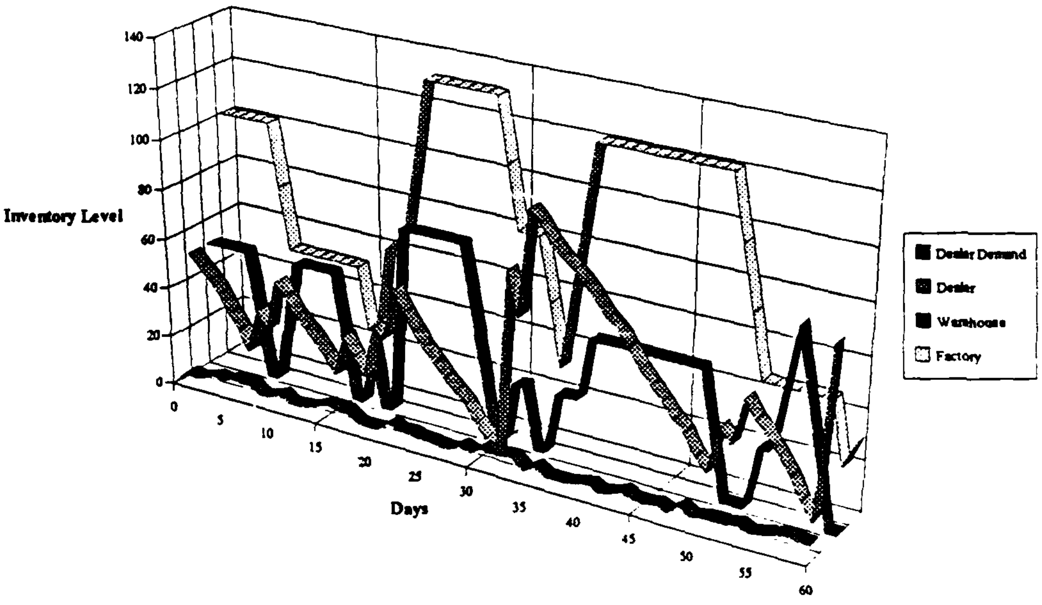


Figure 4.5: Scenario 1—Inventory Level with Probabilistic Demand and Lead Time

2. Three Warehouses, No User Involvement

By adding more warehouses to the scenario, the oscillations in inventory levels permeate throughout the inventory management system and are significantly more pronounced as illustrated in Figure 4.6. The fluctuations are similar to a one warehouse scenario. However, with three warehouses, the increased interdependence compounded with probabilistic lead time result in inventory level rapidly depleting to zero. These low inventory levels cause several t ickorders and a corresponding increase in overall cost for this inventory distribution scenario. So, how should the managers reduce costs?

The complexity of Figure 4.6 fully demonstrates the difficulty of applying traditional analytical techniques to derive an economical solution. By understanding the behavior of the system through simulation with graphical outputs, the manager can develop and test policies in an attempt to reduce cost. The display of inventory levels in Figure 4.6 reveals some inventory patterns. First, all three warehouses rapidly deplete inventory to zero as the dealer and warehouses quickly order to increase inventory. A policy to resolve this dilemma would be for each warehouse to have a higher reorder point. Another apparent pattern is that the factory's inventory is decimated as all three warehouse order simultaneously. Additionally, the factory rapidly cycles through the phases of production. Solutions to these difficulties range from producing larger quantities with corresponding long shutdown periods or by decreasing production rate.

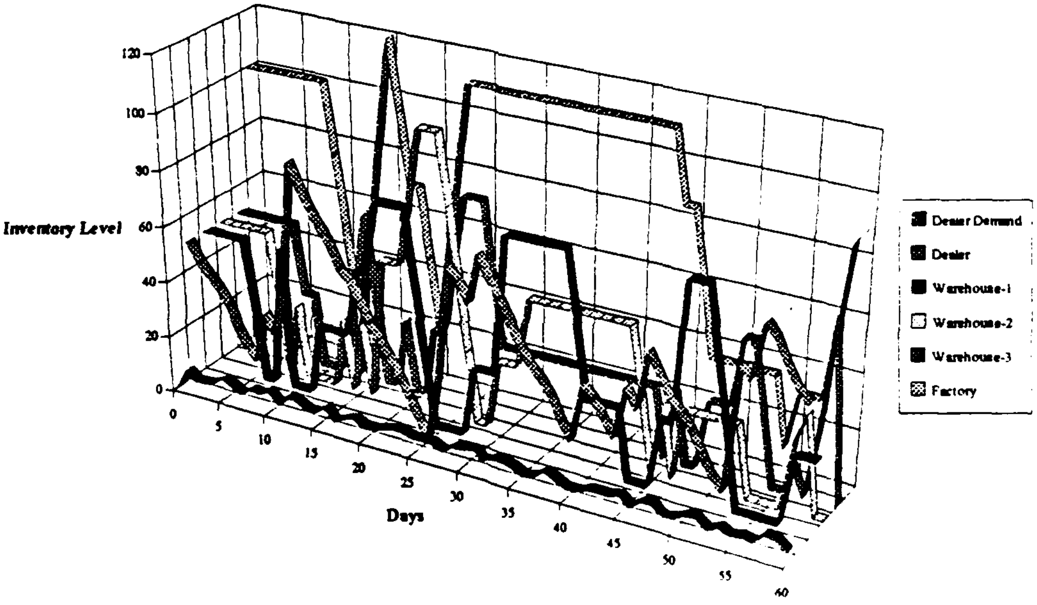


Figure 4.6: Scenario 2—Inventory Levels with Probabilistic Demand and Lead Time

To test these possible remedies, many "what if" analysis can be performed to find the most economical policy for each warehouse, the dealer, and the factory. These particular scenarios assume that order and order points were set and did not change during the simulation. The next scenario will demonstrate the increased complexity involved when each manager attempts to maintain an inventory at a level that mitigates backorders when faced with random demand.

3. Three Warehouses, Full User Involvement

With the exception of ten startup iterations, this scenario differs from the previous scenarios in that the program does not automatically reorder for the dealer or warehouses nor does it automatically start or stop production at the factory. As the simulation iterates through each successive day, the user must make the decision to order or alter factory production based upon inventory level and demand. The dealer and warehouse managers will try to keep their inventories at a sufficient level considering average demand. If inventory levels begin to fall below a desired level the manager will order extra units above his/her sales rate to rebuild the inventory. Conversely, he/she will reduce his/her order rate if inventory level becomes too high. Accurate prediction is limited due to probabilistic demand and lead time. The dealer can estimate demand to stay within a certain range. However, the warehouse managers will find it difficult to predict how much the previous person up the chain will order and when. Additionally, whenever an order is placed it can arrive up to five days later thus giving the potential of zero inventories with corresponding backorders.

The resultant inventory behavior of the three warehouse scenario with user involvement is illustrated in Figure 4.7. Oscillatory behavior is still present but the initial fluctuations are due to the beginning inventories being artificially low for the corresponding demand. Furthermore, the ten startup iterations developed random inventory levels and backorder and do not allow user intervention. The first twenty days demonstrate low to zero inventory levels as the dealer and the warehouses build inventory to approximately one hundred units. Furthermore, inventories do not increase until factory production is sufficient and inventory arrives according to probabilistic lead times.

Once inventory levels are stable, the demand upon the warehouses diminishes. The managers then begin reducing inventory to a more optimum level of approximately eighty. However, warehouse #2 and #3 waited too long and allowed inventory to depleted to zero and incurred momentary backorders. The practice of reducing inventory levels to reduce holding cost without continuous orders leads to the oscillatory behavior. Running this simulation for a greater period would allow each manager to determine an appropriate reorder point and quantity at the least cost.

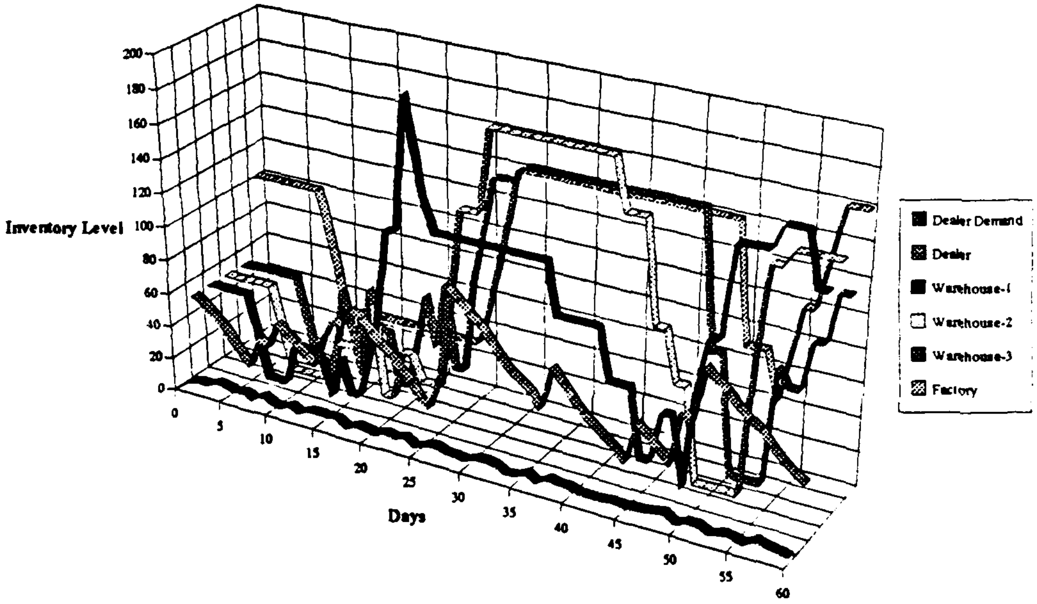


Figure 4.7: Scenario 3—Inventory Levels with Probabilistic Demand and Lead Time

In this scenario, it is cheaper to have an abundant inventory to minimize cost. However, scenarios with increased holding costs will make it more difficult to predict which policies will give the optimal solution. Additionally, a JIT model will illustrate that reduced inventory are more economical. The benefit of the simulation model is to demonstrate the difficulties involved with maintaining an inventory distribution system but allows the managers to experiment with different decision options and evaluate the potential consequences. The three-dimensional graphs illustrate the oscillatory behavior of a distribution system and give the manager an appreciation of the dynamics and interactions involved between each of the individual units.

4. Three Warehouse, JIT Environment

When a JIT environment is simulated, the difficulty in predicting lead time by the manager is mitigated, allowing for more stable inventory levels. An examination of Figure 4.8 reveals that inventory levels remain relatively constant. The occasional jump shown by the dealer and warehouse #3 are a result of lead time being two days vice one. Therefore, inventory level is below the reorder point two days in a row and subsequently two orders are made. The resultant double order by the dealer causes warehouse #1 to face a backorder with its high penalty cost.

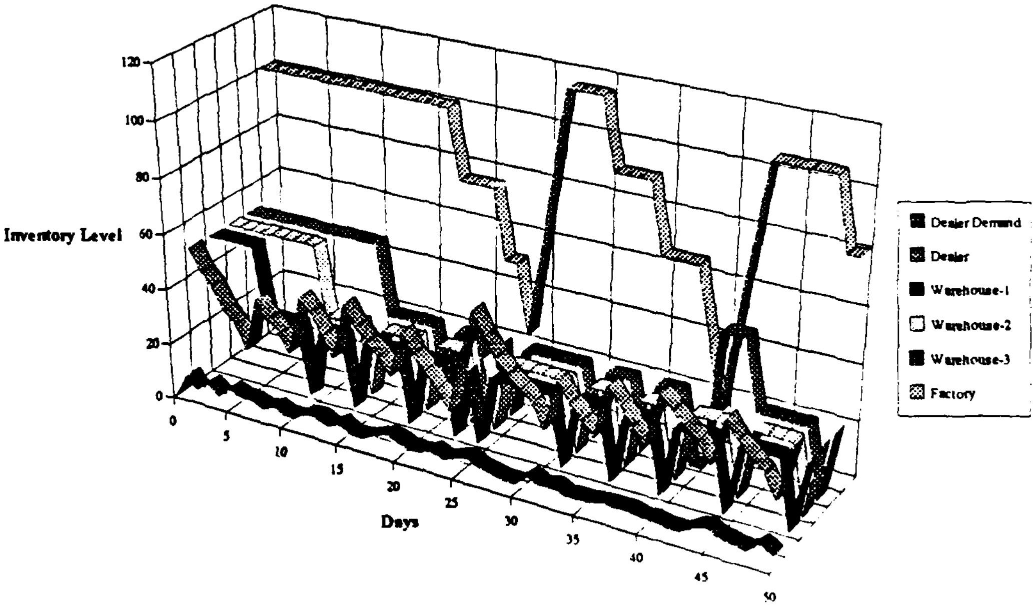


Figure 4.8: Scenario 4: Inventory Level Under JIT

A drawback of the depicted JIT scenario is that the multiple orders cause the overall cost of the system to gradually rise. This could be solved by varying order points and levels to achieve a more economical solution. However, the point of this scenario is to have the same initial data as the other scenarios for comparison. Furthermore, the predictable behavior of Figure 4.8 is the crucial aspect of JIT that should be understood by a manager. The predictability allows a inventory distribution manager to derive a superior inventory policy. This is one of the reasons that many organizations, including Navy Depots, are converting to a JIT inventory distribution system. This is especially the case if holding costs far exceed ordering costs.

Each of the above scenarios illustrate the benefits a manager can derive from spreadsheet simulation modeling of an inventory distribution system. The next chapter will demonstrate how spreadsheet simulation can augment a manger's ability to analyze resource allocation within a queuing environment.

V. QUEUING ENVIRONMENT UTILIZATION

As with inventory management, effective allocation of resources within a queuing environment is another challenge often faced by managers. A queue, or waiting line, is the accumulation of customers, products, etc., at a holding station that are awaiting service or processing within a system. Any institution that provides services or engages in manufacturing and faces the possibility of an item waiting in line deals with some form of queuing utilization. Inventory, for example, is provided to the customer by a server. If a sufficient number of attendants are not available to the customer for delivery of merchandise, excessive queues develop and the customer will seek service elsewhere. The manager may have determined the optimum stock levels required to prevent stockout but his/her analysis would be incomplete if the queue consists of people. If the queue is too long, customers waiting in line would become dissatisfied with the system. The result is lost sales that are the equivalent of stockout costs in an inventory environment. Thus, another facet of a successful manager is the ability to make crucial decisions in the realm of queuing theory.

A. THEORY

Queuing discipline can be approached in many ways with as many corresponding solutions as demonstrated in the previous chapter on inventory management theory. The most rudimentary approach for determining the number of servers or processing stations required for the expected demand is to use trial and error methods. By changing a few parameters and observing the results, a manager can determine the most economical balance between desired service capability and customer waiting time. As the system increases in complexity, it often becomes impossible to achieve an optimal solution through trial and error techniques. Furthermore, if one is designing a new system, such as the number of registers to install in a grocery store, it is not economical to render a guess as to an optimal solution. Therefore, other methods are required such as analytical and simulation techniques for solving more complex scenarios. However, one first must understand some basic aspects of queuing theory as developed by management scientists.

In discussing queues, several common terms are used. A *queue* is that part of the system where units are waiting service. The *server* is the person or device that performs a service to the units waiting in the queue. The *system* itself consists of all queues and servers. *Channels* are the lines within a system that can be simple in nature with a single line or more complex with multiple channels. The number of servers contained within a channel are referred to as *phases*. As with channels, a simple system will have a single phase while more complex systems contain layers of servers or multiple phases. Another term common to queuing systems is *balking*. Balking occurs when queues become too long and the customer chooses not to wait in line. He/she either seeks service elsewhere or comes back at another time. To determine when balking occurs is often difficult to estimate but critical to the design of a queuing system. [Ref. 2:pp. 172-174]

With a knowledge of queuing phraseology, one must also understand queuing discipline. What is the sequence of events within a queue and how are these events distributed with respect to time? Common to most queues is the design of the system that provides service to the first item within the queue or sometimes defined as "first come, first served." Other service philosophies can be used in the design of a system such as assigning categories to items in the queue and serving some categories above others regardless of when they entered the queue. Given a service philosophy, one can assign the proper distribution of events in relation to time. The best approach is to observe the actual system in operation over the period of time that analysis is desired and simulate arrivals to fit the observed pattern. If this option is not available or it is a new system, other models are used within queuing simulations. For arrival rates, a Poisson distribution with its corresponding negative exponential probability of arrival has been found to fit a majority of arrival patterns. If a Poisson arrival distribution is not appropriate, another approach is to model arrivals through a normal or uniform distribution. For service times, observation is the best choice. However, a normal distribution is often used if an observation is unavailable. If a more appropriate distribution function exists, it can be used to model service time.

In choosing the number of servers in a model, one must be careful not to allow arrival rate to approach service rate. As arrival rate nears service rate, an exponential relationship occurs resulting in chaos and the queuing system collapsing upon itself. This potential exponential growth in the queue often does not occur in the real world due to the onset of balking significantly before model failure.

Once the parameters are chosen and the system is properly modeled, the approach offered by management scientists for developing an optimal solution is the application of analytical techniques. As with inventory management, analytical techniques consist of several formulas that provide an understanding of the behavior of the queuing system. Through the behavior of a system, the cost involved in providing the services can be minimized. The costs that must be considered are:

1. *Service costs.* These are the expenses a manager must consider in providing the desired service. These costs include the cost of the material required for providing the service as well as the salary of the employee.
2. *Waiting costs.* These expenses are faced by the manager when units are forced to queue as they await service. These costs include opportunity costs as well as balking costs.

Service costs are often easy to calculate but waiting costs can be much more illusive. Furthermore, waiting costs can sometimes be inappropriate such as with a hospital environment. Instead of placing a value on human life that would occur with balking and possible death, a minimum service level is determined and used to obtain an optimal solution. However, even with a thorough understanding of the queuing formulas, many queuing systems are too complex to be solved using traditional analytical techniques. Thus, one must pursue simulation methods to derive an optimal solution.

B. QUEUING SYSTEM SIMULATION

The queuing system that was simulated using *Excel 4.0* is a highly complex analysis of a hospital emergency room. The hospital emergency room that was modeled includes multiple phases and multiple channels. The complexity is further compounded by using a modified "first in, first serviced" philosophy by accounting for patients who need immediate care. The arrival rate of patients employes a Poisson distribution that varies over time. The number of nurses and doctors in the emergency room for treating patients varies during time to deal with changing levels of staff workload during a day. Additionally, five different patient types are considered to model varying levels of treatment and service rates by the nurses and doctors.

To solve this queuing problem and multiple "what if" scenarios, one could use analytical techniques. However, deriving an analytical solution would consume vast amounts of time. An analysis of a one week scenario using the model developed took approximately thirty minutes on a high speed personal computer. Performing several "what if" scenarios consumed several hours and illustrates the benefit of computer simulation. Furthermore, as with the inventory model, several parameters can be manipulated within the model to simulate different scenarios. A full listing of the program with comments is provided in Appendix B.

The primary challenge in the design of a spreadsheet queuing model is how to account for the sequence of events in relation to time. Unlike the inventory model, time is not a fixed event to which all other events are linked. With a queuing system, time is a variable that is controlled by all other events and is governed by a "next event" discipline. For each iteration, the model does not increment time to determine what happened during the fixed time unit but instead asks the question, "What happens next?". To use the technique of "next event" timing, the model iterates time backward instead of forward. Each event is assigned a completion time that is incremented to zero by another event's completion. The event whose time is closest to zero is the next event to which action occurs and triggers all other events.

To catalog events and their times of action, events are classified as either primary or secondary. A primary event causes action to occur in the model such as arrival of a unit or the completion of service. Secondary events, such as entering or leaving a queue, are a direct result of primary events. Additionally, the occurrences of primary and secondary events result in the scheduling of other primary and secondary events in a chain-like manner. To fully understand this concept requires the use of a flow chart that depicts the passage of a patient through the emergency room as illustrated in Figure 5 1 Flow charts

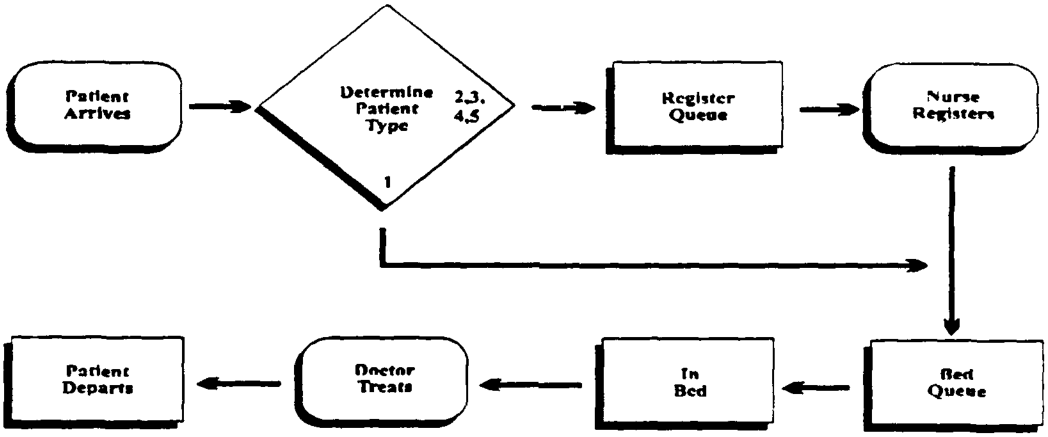


Figure 5.1: Chain of Events for a Patient

are also provided for each primary event with its corresponding chain of secondary events. Rounded rectangles within Figure 5.1 contain the three primary events that occurr within the the hospital emergency room. The three primary events that can occur with each patient are the arrival to the emergency room, the completion of registration by a nurse and the completion of treatment by a doctor. Several secondary events occur as a result of these primary events but first a discussion of Figure 5.1.

Figure 5.1 shows the chain of events that occur for each individual patient within the hospital emergency room. The first primary event to occur is his/her arrival. The chain of events for the primary event of a patient arrival is illustrated in Figure 5.2. A Poisson arrival distribution was used to simulate arrival rate. Other arrival distributions could be modeled through the altering of a few programming lines in the spreadsheet macro. For

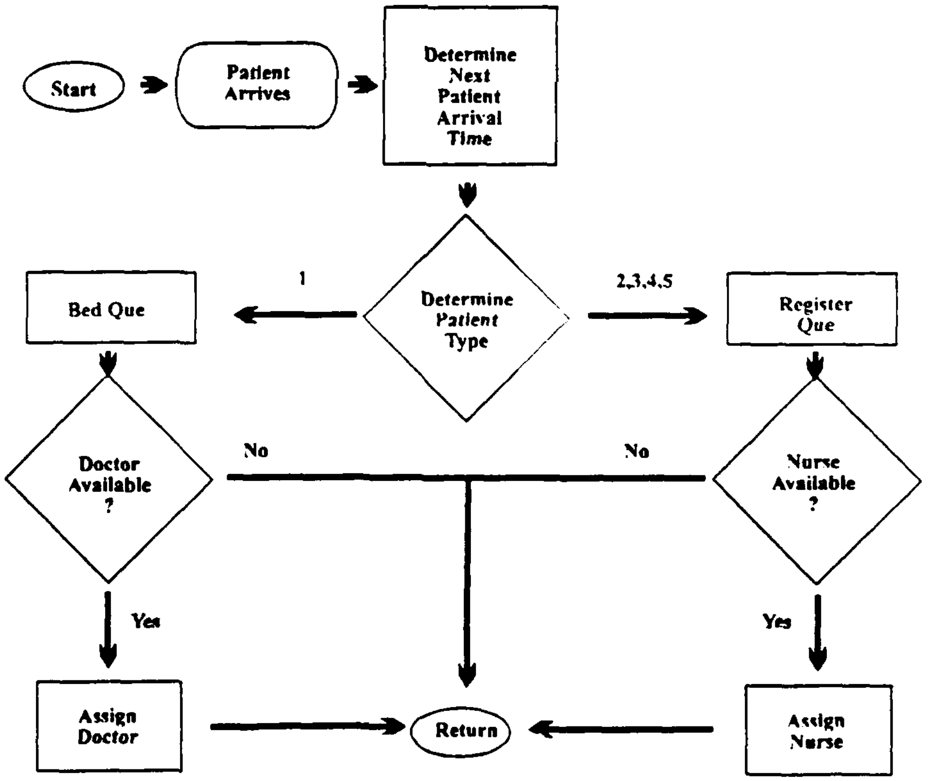


Figure 5.2: Primary Event—Patient Arrival

"next event" time philosophy, a secondary event is to calculate the arrival of the next patient. Each arrival to the emergency room was calculated using a negative exponential distribution. The parameters for the calculation are time of day and mean arrival rate. This calculated arrival time is used as a count down time for the next patient's arrival. Another secondary event is the determination of the patient medical classification or type. This value determines if the patient goes straight to the bed queue (open wounds, i.e., type 1) or to the registration queue (less sever cases, i.e., types 2 - 5). Within each respective queue, a determination is made as to whether a nurse or doctor is available respectfully. If either is available, the patient is removed from the queue and is assigned to a server. The nurse or doctor is then assigned a normal distribution treatment time based upon patient type.

The next primary event to occur is that of a nurse completing registration of a patient with its corresponding secondary events as illustrated in Figure 5.3. The first secondary

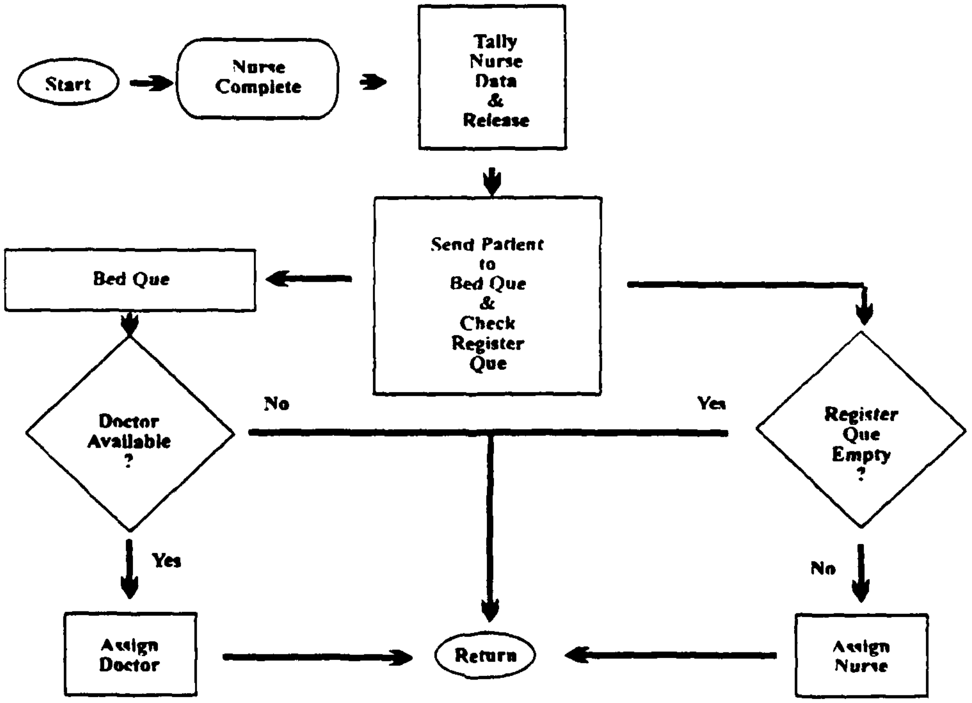


Figure 5.3: Primary Event—Nurse Completion

event is to collect data on the nurse's service (i.e., time spent with patient, etc.) followed by a freeing of the nurse for further patients. The registered patient is sent to the bed queue and is assigned a doctor if one is available. The free nurse is assigned to another patient if one exists in the registration queue. If assigned, a service time is determined as previously discussed. The service time is used for the next event analysis.

The final primary event within the model is the completion of service to the patient by the doctor. The secondary events are shown in Figure 5.4 with the first event being the

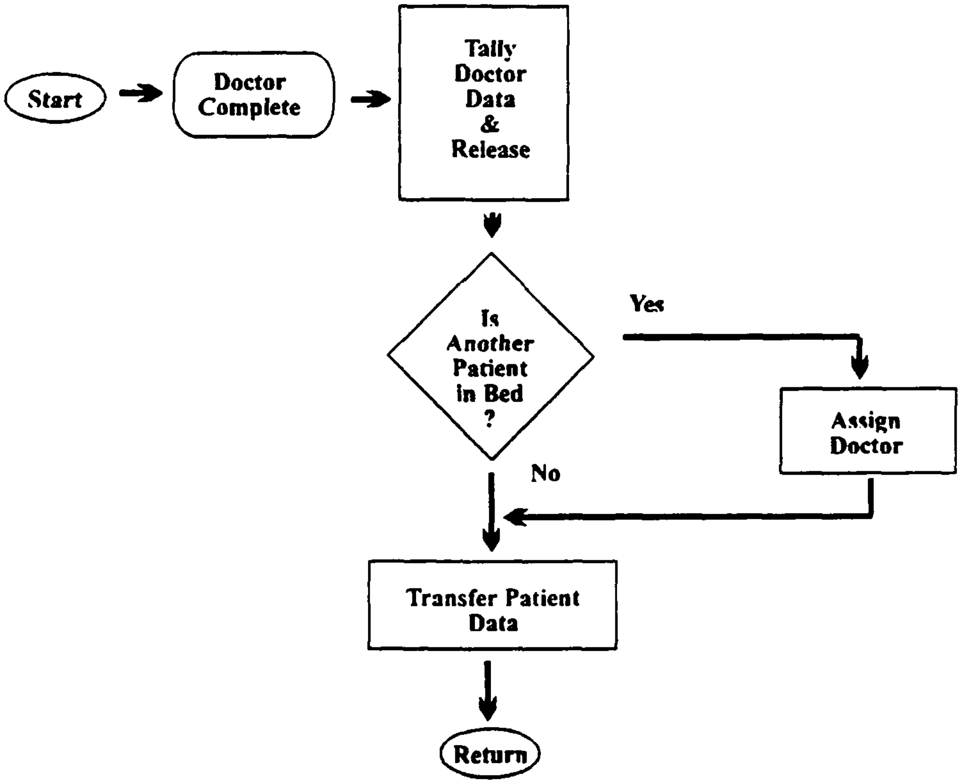


Figure 5.4: Primary Event—Doctor Completion

tallying of doctor statistics. The doctor is then freed for treating further patients. The information on the treated patient is transferred to a location of the spreadsheet for later analysis. The final event is to check if another patient is in bed who needs treatment. If so, the doctor is assigned a completion time as previously discussed with the time being used for next event analysis.

By understanding the primary and secondary events within the hospital emergency room, a spreadsheet simulation model can be developed. Similar to the inventory model, one must first establish the objectives of the simulation. The primary objective of the model is to collect a number of statistics that can be analyzed by the user to determine the behavior of the system and facilitate human resource (i.e., doctors and nurses) allocated and assignment decission. Another objective is to provide a number of options to the user to accommodate "what if" analysis. The spreadsheet model that was developed allows a number of different choices for optimization of hospital queuing problems. Choices vary from the number of nurses and doctors available (up to four each) with their respective shifts to the number of beds available (up to ten). With the provided statistics and by varying the available parameters, an optimal solution can be obtained.

Some of the aspects of queuing discipline are not built into this model but could if so desired by changing a few lines within the macro. Balking was not addressed since few people have the choice of multiple hospitals or the ability to leave. Additionally, since this is a service scenario where lost sales are not the concern but instead adequate services is, costs were not analyzed. The purpose of the model is to determine the appropriate service time taking into consideration the patient load by varying the number of beds, nurses, and doctors.

Before demonstrating the capabilities of the queuing simulation model, a few key differences exist between the queuing model and the inventory distribution model and should be understood. The first key difference is how time was managed as previously discussed. Second, since the queuing problem does not contain separate levels of management (i.e., factory, warehouse, and dealer), the spreadsheet model did not initially lend itself to the creation of completely separate macros. Instead, one large macro was programmed with several subroutines nested within the one macro. If desired, each subroutine could be developed into a separate macro. However, during the initial development it was found easier to work within one macro. As the complexity of the model was increased, the size of the individual macro grew. Towards the completion of the model, some of the advantages to a single macro were lost such as the inability to further name cells or use *Excel's* macro debug add-in. Therefore, if designing a more complex model or modifying this one, it would be advantageous to separate the macro into smaller macros.

To begin all scenarios, the user integrates with the queuing macro through a master worksheet in the same fashion as the inventory model. Dialog boxes will ask for initial input and the model will begin simulation. All pertinent data is transferred to the worksheet at the completion of the simulation.

C. SCENARIOS

With an understanding of queuing philosophy and "next event" time management, two scenarios will be simulated to demonstrate the full capability of the spreadsheet simulation model that was developed using *Excel 4.0*. Each will illustrate the benefits of using simulation for determining an optimal solution in a complex queuing environment in a hospital emergency room. Even if an optimal solution is not desired, a manager can pose "what if" scenarios and study the behavior of the queuing system. He/she can than make changes to the system to meet his/her desired objectives.

Without using simulation, a manager or director of a hospital emergency room would have few options for determining how many nurses, doctors, and beds are required to meet the anticipated patients' arrival rate. One option would be trial and error. His/her first iteration would be to compensate the demand by over-staffing the emergency room with doctors and nurses for several weeks. After several weeks. sufficient data would accumulate allowing the manager to make further iterations by reducing staff size. His/her primary goal is the providing of adequate service with minimum patient waiting over a twenty-four hour period. After several months, the manager will achieve the service he/she desires. Unfortunately, with a changing environment and the practice of fluctuating staff size being economically unfeasible, the manager will view trial and error methods as a less than optimal approach. The following scenarios will illustrate how simulation is a far superior method than trial and error.

The first scenario will analyze the aforementioned trial and error method in that the manager approaches the problem by over-staffing the emergency room. During a one week period, ten beds, four nurses, and four doctors will be available full time to treat the patients in the emergency room. This simulation will show a gross under-utilization of the facilities. However, the simulation will provide the best service available and can be used as a point for other simulations. The second simulation will be a compilation of several one week simulations. Many "what if" scenarios will be simulated by varying the number of servers and beds available. Each change will be based upon the utilization and queuing results of previous scenarios. The goal of the second set of simulations is to obtain an optimal balance between adequate service and server utilization in a hospital emergency room. Also, another criterion will be to ensure that few patients wait for the use of a bed.

For all scenarios, the patient arrival rate was based upon a Poisson distribution with mean arrival rate varying during the day as shown in Figure 5.5. Treatment time was the same for each nurse and doctor and modeled upon a uniform distribution. The distribution of time varied by patient's category with a lower, upper, and standard deviation as shown in Table 5.1. Within the spreadsheet model, all of these values are inputted by the user and can be changed to reflect actual data. To allow comparability between scenarios, only the numbers of beds, nurses, and doctors are altered.

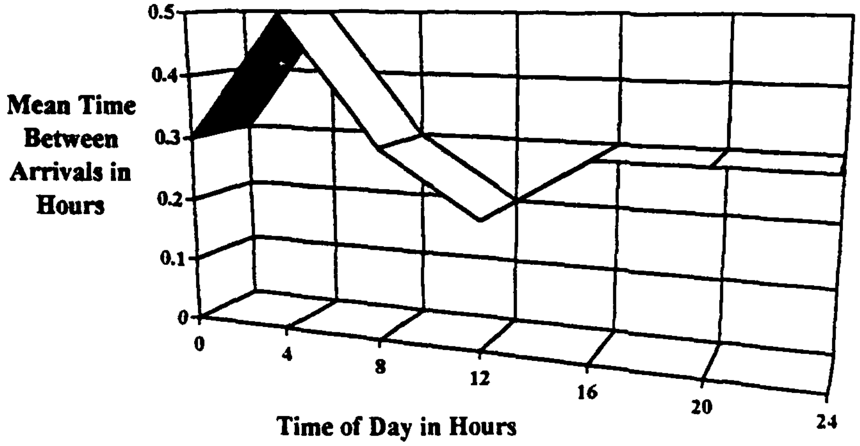


Figure 5.5: Patient Arrival Distribution

TABLE 5.1: PATIENT CATAGORIES AND TIME DISTRIBUTIONS

Patient Category	Description	% of Total	Lower	Upper	Standard Deviation
1	Open Wounds	8	0.25	0.35	1.0
2	Closed Injuries	13	0.20	0.30	1.0
3	Multiple Trauma	33	0.15	0.25	1.0
4	Visceral Complaints	20	0.10	0.20	1.0
5	Chronic Complaints	26	0.05	0.15	1.0

1. Full Services with No Optimization

As one would expect, under a full service environment with the given patient arrival rate as depicted in Figure 5.5, an under-utilization of the emergency room's facilities occurs as shown in Table 5.2. To understand the utilization rate of the emergency room facilities, one must understand the simulation model. The model is designed to seek the nurse or doctor who has been without a patient for the longest amount of time resulting in the patient load being relatively evenly distributed among the staff. Having staff only utilized less the 19% of the time is not economical to the hospital. For bed utilization, the model tries to fill whichever bed in not in use starting with the first bed and progressing to the tenth bed. Therefore, from Table 5.2, one can observe that only five beds are required with the fourth and fifth being used less than 1%. Table 5.3 provides other pertinent statistics of the full service queuing model . All show gross under-utilization of facilities.

TABLE 5.2: SCENARIO 1—SERVER UTILIZATION

Nurse	Utilization	Doctor	Utilization	Bed	Utilization
1	17.65%	1	14.79%	1	38.37%
2	7.03%	2	12.96%	2	19.07%
3	11.75%	3	10.66%	3	5.88%
4	11.49%	4	13.39%	4	0.82%
				5	0.13%
				6,7,8,9,10	0.00%

TABLE 5.3: SCENARIO 1—SYSTEM UTILIZATION

Average Patients in System	1.125 Patients
Maximum Time in System	28.53 Minutes
Average Registration Queue Length	0.00 Patients
Average Time in Registration Queue	0.00 Minutes
Maximum Time in Registration Queue	0.00 Minutes
Percent Who Wait for Registration	0.00 %
Average Bed Queue Length	0.00 Patients
Average Time in Bed Queue	0.00 Minutes
Maximum Time in Bed Queue	0.00 Minutes
Percent Who Wait for a Bed	0.00 %

From this simulation, the manager can understand much about the behavior of the system that will be used in further simulations. First, reduce the number of beds available from ten to at least five. One must be careful though because as the number of servers is reduced, the bed usage will correspondingly increase. The second behavior is the utilization of nurses and doctors. From this simulation, a manager can reduce the available resources by at least one nurse and one doctor. Ideally, he/she can reduce the staff to one nurse and doctor during off hours with augmentation of two or three of each during peak hours. All of these "what ifs" will be analyzed in the next section.

2. Limited Service with Optimization

The following scenario is a product of several "what if" scenarios. This process took several hours of computer time but it is still significantly more economical than trial and error methods over several months. The final values of utilization for nurses, doctors, and beds are based upon the author's judgment of what is believed to be an optimal solution. A professional hospital administrator would be able to apply this model to an actual hospital emergency room to which he/she could achieve an optimal solution based upon his/her expertise.

The final number of nurses, doctors, and beds that were modeled was two, two and five respectively. A nurse and doctor were available twenty-four hours a day while a second nurse and doctor were assigned during peak hours from 0800 to 1600. The five beds were available during the entire twenty-four hours. Table 5.4 presents the utilization of the emergency room facilities. The nurse's and doctor's utilization increases significantly from the first scenario but they are not over-utilized. Table 5.5 presents the statistics for the queuing environment. The length of time in the system along with the number of patients in the system increased but not to unreasonable values. Additionally, nobody had to wait for a bed and that was one of the criterion of the system.

TABLE 5.4: SCENARIO 2—SERVER UTILIZATION

Nurse	Utilization	Doctor	Utilization	Bed	Utilization
1	38.62%	1	40.35%	1	42.78%
2	**30.70%**	2	**33.89%**	2	**23.31%**
				3	6.71%
				4	**1.14%**
				5	0.36%

TABLE 5.5: SCENARIO 2—SYSTEM UTILIZATION

Average Patients in System	1.354 Patients
Maximum Time in System	**72.05 Minutes**
Average Registration Queue Length	0.106 Patients
Average Time in Registration Queue	**1.992 Minutes**
Maximum Time in Registration Queue	35.800 Minutes
Percent Who Wait for Registration	**31.21 %**
Average Bed Queue Length	0.00 Patients
Average Time in Bed Queue	**0.00 Minutes**
Maximum Time in Bed Queue	0.00 Minutes
Percent Who Wait for a Bed	**0.00 %**

Even with the above results, a manager must understand how those results are derived. These values are based upon a one week scenario derived from probabilistic data. Therefore, there is a range of randomness that exists which can cause some values to fluctuate. In Figure 5.6, the number of total patients in the system is illustrated. A number of peaks exist which do not necessarily correspond to the patient arrival rate as illustrated in Figure 5.5 or the number of nurses and doctors available. With probabilistic scenarios, patients will sometimes arrive only a few minutes apart leading to a temporary overload of the servers resulting in queues. This is the cause for a number of the peaks shown above.

Therefore, like an inventory management system, a hospital manager can include safety buffers within his/her ideal solution. Extra beds could be positioned in the emergency room and nurses and doctors can be put on call to handle unexpected increased workloads.

Even with an understanding that simulation methods do not produce a perfect answer, simulation does provide guidelines and an understanding of the behavior of a queuing system. Instead of the expensive approach of trial and error, a manager using a queuing system can apply simple modeling techniques to understand the behavior of a system and arrive at a plausible solution. Juxtaposing the solution with the manager's professional expertise will result in the optimal policy for a queuing environment. The next chapter will demonstrate how a manager can not only use spreadsheet simulation for inventory and queuing problems but also for approaching complex financial dilemmas.

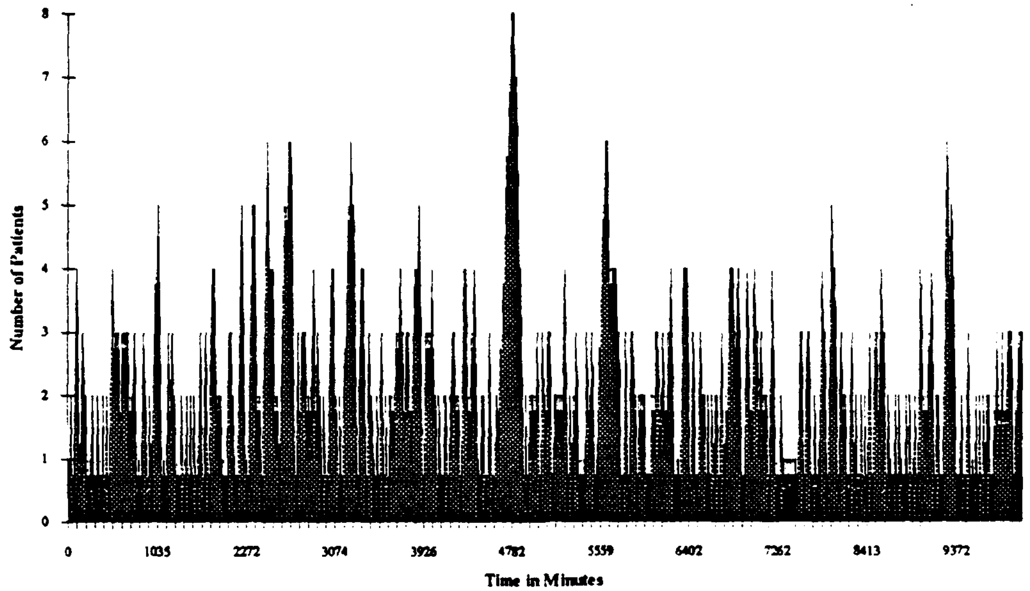


Figure 5.6: Patients in the Hospital Emergency Room

VI. FINANCIAL BUDGETING

The previous two chapters focused on how spreadsheet simulation can be employed to resolve two very specific resource allocation predicaments: inventory distribution and queuing utilization. Not all managers will be confronted with either of these unique management dilemmas. However, virtually every manager is faced with the prospect of justifying his/her need for financial resources in the future. In corporate America and the Department of Defense, financial capital is a scarce resource that must be pleaded for and then allocated within an organization. The allocation justification process is an estimate, often itemized, of expected income and expense for a given period in the future or more succinctly known as financial budgeting. Estimation is the relevant term in the definition of budgeting that lends financial budgeting as an apropos subject for the discipline of simulation. Thus, this chapter will focus on resource allocation as it pertains to financial budgeting which is another facet of a successful manager's prowess.

A. THEORY

The creation of a financial budget is an integral part of any organization that will be confronted by every manager who is responsible for financial capital. The manager`s performance is frequently judged upon his/her ability to submit a budget to management and then stay within its limits. Therefore, the methods employed by a manager when designing his/her segment of the budget is crucial to his/her viability as well as to the organization. Unfortunately, many aspects of budget formulation are not easily quantified. Interrelationships among cash flow items are not always clear, the operating environment is often turbulent, and predicting the future based upon historical data, trends, and managerial judgment are just a few of the factors that must be considered when a manager takes on the task of budget formulation. So how does a manager create a budget and how can spreadsheet simulation simplify his/her quandary?

This chapter will assume that the reader understands basic budgeting methodology as only a few budgeting concepts that are relevant to the discussion will be introduced. There are essentially three levels of budgeting complexity [Ref. 2:p. 80]. The first is the most simplistic and is the foundation for the other two levels. This foundation level of budgeting is applicable to situations that entail little uncertainty in quantifying income or expense and in predicting the future. The manager will often employ a spreadsheet to develop the budget. He/she will categorize income and expenses onto the spreadsheet and project into the future. The summation or bottom line for the year constitutes the manager's budget. Therefore, the manager is using simple deterministic estimates to create a budget. This is called line item budgeting in the business world. This method is frequently appropriate for many situations within a business. However, with more factors considered or a larger fraction of the business included in the budget, the line item budget becomes too inaccurate. Management will then cross into the second level of budgeting.

The second level of budgeting increases in complexity because it employs mathematical relationships to quantify interactions between variables. By identifying casual relationships, the manager accounts for important functional relationships among variables and significantly increases the accuracy of his/her budget. Mathematical relationships between different aspects of a business vary but are often segmented into fixed and variable costs. Fixed costs are similar to values used in line item budgeting. They are those that are part of the production expense that will occur even if no products are produced such as equipment depreciation. Thus, fixed costs do not vary during the year and are easily quantified. However, with variable costs, the manager's experience becomes a factor. An example of a variable cost is the expense of producing one product such as the materials within the item. The manager's experience is required for variable costs because he/she must predict how many products will be produced. However, similar to line item budgeting the values that are included in formulation of the budget are still deterministic. Thus, the prediction about the future fails to quantify the uncertainty relating to the future. This is the key weakness of this method of budgeting. To resolve this dilemma, a manager will evoke the third level of budgeting.

The third level of budgeting is the most complex and is the point where spreadsheet simulation becomes essential to the solution. The third level employs probabilistic techniques to quantify uncertainty associated with future events. The remainder of this chapter will focus on how a second level spreadsheet budget can be augmented by simulation for a more realistic budget. However, there will never be a 100% correct answer because one can never fully predict the future.

To understand how simulation can be of benefit to the manager in the budgeting process, one must first be introduced to *Monte Carlo* methodology. The Monte Carlo technique can be defined as a method where a stochastic variable is assigned a value for use in a calculation by drawing a random value that is correlated to the probability distribution of the variable.

The Monte Carlo concept is best comprehended by a simplified example. A manager is requested to estimate sales of a particular product based upon his/her collective experience. He/she provides an optimistic prediction of 1000 units and pessimistic value of 500 units with a uniform distribution between these extremes. From this information, a uniform distribution is created that can be used in a Monte Carlo simulation. The pessimistic value is assigned a value of zero while the optimistic value is assigned a value of one. A random number is then generated between zero and one. If the random number was 0.3, a ratio between 500 and 1000 units will results in 650 units of sales.

The above example is a simplistic example of the Monte Carlo method but it illustrates the essential steps. First, a probability distribution is assigned to the variable of interest. There are several distributions to choose from such as uniform, triangular, normal, Poisson, and tabular to name a few. What distribution is best must be determined by collecting previous data that is normally surmised into a frequency distribution that can be correlated to an appropriate distribution. An appropriate caution must be stated before preceding any further. A significant assumption has been made when assigning a probability distribution based upon historical data. Monte Carlo methods are founded upon the assumption that the historical data with its corresponding distribution are in fact a true representation of the variable's interaction in the past and can be projected into the future. Otherwise, the model will produce results that are misleading and of no benefit to the analysts.

If the variable of interest is not quantifiable with past data, the probability distribution must originate from the manager. He/she must either estimate what he/she believes are the pessimistic, optimistic, and most likely values or chance occurrence for each range of possible values for each variable. This is not an easy task but it surpasses the alternative methods of intuition or trial and error.

The second step of the Monte Carlo method is to assign the probability distribution a corresponding range of values from zero to one. The third step is to generate a random number from zero to one that is then applied to the probability distribution for a corresponding value that is then used in further calculations.

The fourth step is to perform steps three and four several times to allow the law of averages to work. The result is a range of values for the value of interest or objective variable such as total budget, net present value, etc. This range of values is then tabulated into a frequency distribution from which further analysis can be performed The frequency distribution is the paramount benefit of the Monte Carlo technique. Instead of one deterministic result, a range of values is presented to the manager that give the probability of achieving the objective. Thus, the manager can make a budgetary decision with significantly more confidence than the deterministic approach.

Thus, Monte Carlo methods offer several benefits to the manager when he/she is confronted with a complex financial budgeting dilemma. He/she can assign due weight to uncertain quantities and relationships vice relying totally on one deterministic value. Additionally, he/she has more information from which to base his/her decisions. Furthermore, Monte Carlo methods are relatively inexpensive to perform and can be easily adapted to a computer spreadsheet from which most budgets have their origins. The next section will discuss how a spreadsheet can be programmed to perform Monte Carlo simulation.

B. *MONTE CARLO SIMULATION*

A computer spreadsheet is relatively easily programmed through a macro to employ the Monte Carlo methods. The macro that was created with *Excel 4.0* is much less intricate than the two previous chapter's macros and required the least amount of time to create. The only real difficulty encountered during programming was the creation of a method for the macro to be dynamically linked to the worksheet through variable names and cell reference without explicitly using the worksheet's name so as to make the macro applicable to any appropriate spreadsheet. Therefore, the Monte Carlo macro is fundamentally different from the inventory distribution or queuing utilization macros and can be applied to any spreadsheet that has probabilistic parameters with a desired objective. The macro has more applications than just financial budgeting and this makes it the most universal of the three macros developed during this study.

A caveat must be understood before discussing the model. Monte Carlo simulation is not simulation in the same sense as inventory or queuing simulation. When a manager employs Monte Carlo methods, the results of the simulation are a frequency distribution of possible eventual outcomes that can be analyzed and used for decision analysis. However, this method does not demonstrate the *behavior* of the system as it changes over time which is the essential element of the previous two chapters. Therefore, Monte Carlo simulation is not true simulation but instead a sampling technique for determining probabilistic values for a crucial variable. These values can then be used in a true simulation model for a more realistic behavior analysis.

The development of the Monte Carlo macro begins with a flow chart that identifies key interactions within the model. Figure 6.1 depicts the events in the macro that begins with the identification of essential parameters and retrieval of data from the worksheet if it was previously saved. The next chain of events is to assign probabilistic variables on the spreadsheet with a respective probability distribution. Four distributions are offered in the macro: uniform, triangular, normal, and tabular. *Excel 4.0* is capable of several other distributions that can be programmed into the macro if so desired.

The next sequence of events is the heart of the Monte Carlo method. A random number is generated for each variable that is used to determine the variable's value from its corresponding probability distribution. After all variables have been assigned a value, the objective variable is recorded and relative frequency data is tabulated. This process repeats itself for the number of iterations chosen by the analyst.

Once the iterations are complete, the model transfers input data and the output statistics for further analysis to the worksheet if desired by the analyst. The worksheet is also returned to its original state by changing the manipulated variables to their original values. The Monte Carlo analysis of a worksheet is then complete.

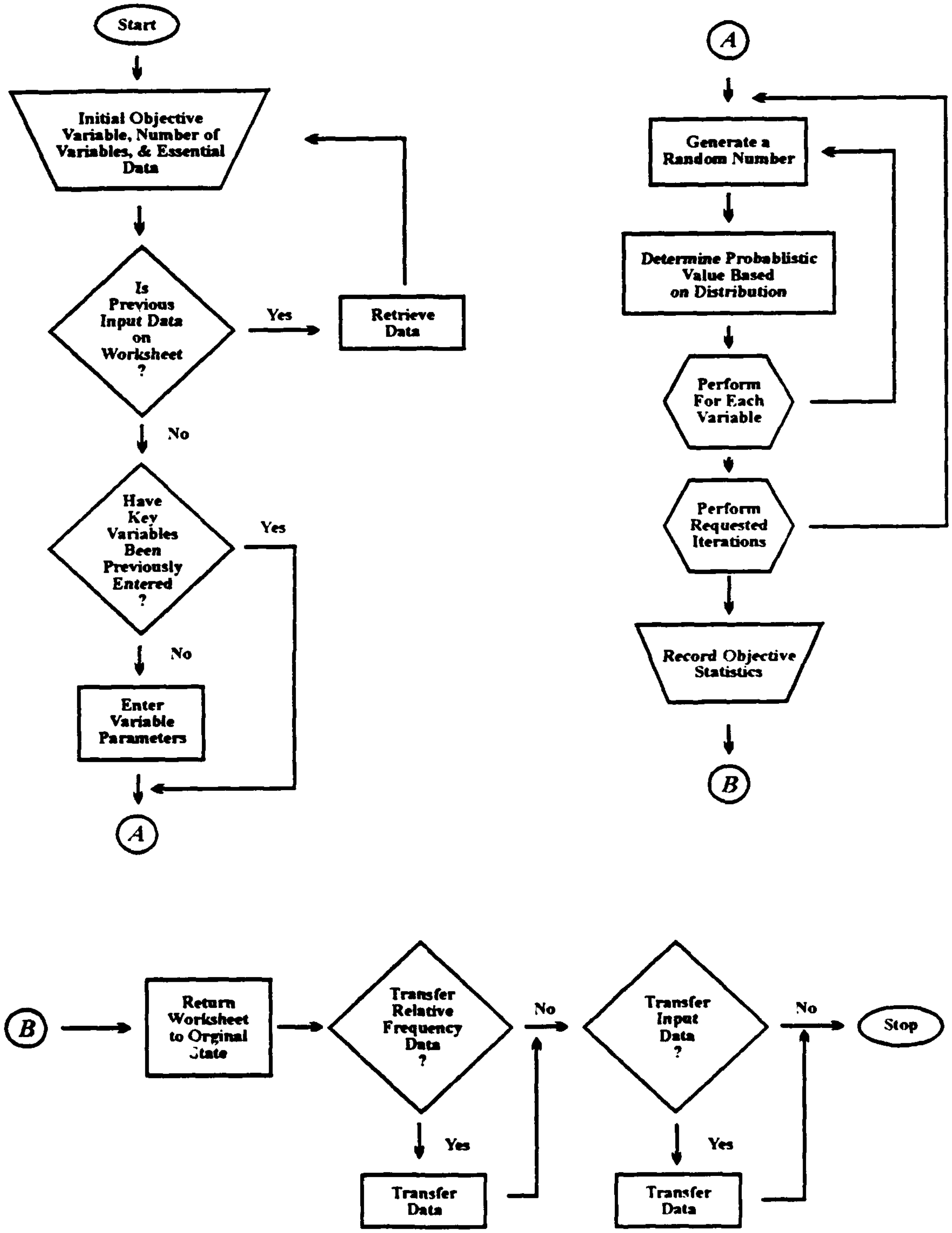

Start
Initial Objective Variable, Number of Variables, & Essential Data
Is Previous Input Data on Worksheet ?
Yes
Retrieve Data
No
Have Key Variables Been Previously Entered ?
Yes
No
Enter Variable Parameters
A
A
Generate a Random Number
Determine Probablistic Value Based on Distribution
Perform For Each Variable
Perform Requested Iterations
Record Objective Statistics
B
B
Return Worksheet to Orginal State
Transfer Relative Frequency Data ?
No
Yes
Transfer Data
Transfer Input Data ?
No
Yes
Transfer Data
Stop

Figure 6.1: Monte Carlo Flow Chart

Key to the accuracy of the simulation is the number of iterations that were performed by the macro. There are no rules of thumb or mathematical methods for determining the number of iterations required to achieve accurate results. However, a method can be employed to ensure that results are consistent. First, the simulation is performed with a few iterations around a wide range of possible objective results. The resultant upper and lower limit from this limited simulation is then used in the next simulation run in which significantly more iterations are performed. A third run using more iterations is then compared to the second run. The frequency distribution comparison of each run should not be significantly different when plotted on the same axis. If they are, more iterations are required until the distributions are relatively equal.

With a Monte Carlo macro developed, the manager is ready to embark upon financial budgeting using Monte Carlo methods. To apply the macro, a few concepts must be built into the worksheet which contains the budget information. First, all essential variables that affect the objective must be included on the worksheet. Second, for each essential variable, it must be classified as either state, policy, or environmental. State and policy variables are not pertinent to the Monte Carlo macro. Environmental variables are the variables for which a probability distribution must be determined and modeled by the Monte Carlo method. The third step is to create the worksheet based on the previous two steps. Essential to the creation of the worksheet is to ensure that all variables and the objective are linked by formulas. Thus, as the environmental variables are changed by the macro, the objective also changes and statistics can be generated.

The above concepts are required for the proper execution of the Monte Carlo macro. The macro will perform the simulation with any worksheet designed in this manner. It does not have to be a financial budgeting scenario. Also, if one understands how the macro interacts with the worksheet, he/she can speed up the initial variable entry process. The macro communicates to the user through *dialog boxes.* For each variable, a dialog box requests either the variable's cell location or name. The name function on the dialog box provides a list of all names for the worksheet. Thus, by naming all the environmental variables, the analyst does not have to search though the worksheet to locate a cell reference.

The Monte Carlo method and macro have now been introduced. To fully appreciate the benefits that can be rendered by these techniques, a financial budgeting scenario is modeled and simulated in the next section.

C. SCENARIO

The Department of Defense application of the Monte Carlo simulation method that was analyzed was the 1974 through 1976 budget of the Civilian Health and Medical Program for the Uniformed Services (CHAMPUS). All of the information that will be presented is derived from Maassen and Whipple [Ref. 4] and the following analysis does not attempt to verify or repute the reported results. Additionally, some simplifications were made to the worksheet so as not to make the model too complex.

Illustration 6.1 shows the budget for CHAMPUS as estimated for 1974 through 1976. A simplification of the derivation of the values included in Illustration 6.1 is present in Illustration 6.2. The 54 shaded areas on Illustration 6.2 are the environmental variables that are linked to Illustration 6.1 by the italicized variables. They affect the total budgeted obligation on Illustration 6.1 which is the objective value. The methods which the Navy used to derive these values are rudimentary and therefore lead one to question their accuracy. Additionally, the derivation of some of the values was not explained by Maassen and Whipple [Ref. 4].

For the straight line projection environmental variables, the values are based upon a determination of the percentage change that occurred during the previous year, 1973. No

Department of the Navy —Medical Care in Non_Service Facilities

	Average Daily Patients			Cost Per Day			Total Obligations (in 1000's)		
	FY 1974 Estimate	FY 1975 Estimate	FY 1976 Estimate	FY 1974 Estimate	FY 1975 Estimate	FY 1976 Estimate	FY 1974 Estimate	FY 1975 Estimate	FY 1976 Estimate
Inpatient Care									
Active Duty Dependents	1,526	1,517	1,493	$136.07	$156.48	$156.48	75,777	86,653	85,298
Retired Dependents	1,323	1,407	1,470	$79.18	$91.06	$91.06	38,223	46,752	48,861
Retired Personnel	386	426	445	$98.29	$113.03	$113.03	13,843	17,555	18,348
Cost Sharing Reduction								(8,529)	(8,706)
Subtotal, Inpatient Care	3,234	3,349	3,408				127,843	142,432	143,801
Outpatient Care									
Active Duty Dependents	620	590	575	$18.13	$20.85	$20.85	4,103	4,490	4,376
Retired Dependents	1,025	1,148	1,200	$15.86	$18.24	$18.24	5,935	7,643	7,989
Retired Personnel	274	326	341	$19.99	$22.99	$22.99	1,999	2,735	2,861
Add Cost Sharing								3032	3058
Subtotal, Outpatient Care	1,919	2,064	2,116				12,036	17,900	18,284
Outpatient Care (Pyschotherapy)									
Active Duty Dependents	715	378	373	$23.80	$27.37	$27.37	6,211	3,776	3,726
Retired Dependents	423	242	253	$21.69	$24.94	$24.94	3,349	2,203	2,303
Retired Personnel	94	98	102	$22.34	$25.69	$25.69	767	919	956
Reduction for Psychiatric Care									
Subtotal, Outpatient Care (Psychotherapy)	1,232	718	728				10,326	6,898	6,986
Drugs							2,428	2,887	2,945
Retarded and Handicapped							3,215	1,854	1,891
Dental							4,182	747	0
Contractors' and Services; Other Costs							6,002	6,478	6,539
Service Population Estimates							50	1,300	0
Quality Control Programs							0	359	0
Estimated Contractual Care Program Requirements per Year							$166,082	$180,855	$180,446
Total Contractual Care Program Requirements									$527,383

Illustration 6.1: CHAMPUS Budget 1974 — 1976

Average Daily Patients (ADP)

Inpatient Care

	Previous Year	Straight-line	Projected	Adjustments			Projection	Various	Population		Projection	Various	Population	Projection
				Contractor	Boston	Doctor								
	1973	Factor	Estimate	Backlog	Closure	Shortage	for 1974	Adjustments	FY74	FY75	for 1975	Adjustments	FY76	for 1976
Active Duty Dependents	1146	1.286	1,474	43		9	1,526	(18)	902,969	908,609	1,517	(4)	896,762	1,493
Retired Dependents	920	1.393	1,282	30	3	8	1,323	10	824,250	870,088	1,407	0	909,335	1,470
Retired Personnel	261	1.421	371	9	4	2	380	17	311,754	329,277	426	0	344,147	445

Outpatient Care

(other than psychotherapy)	Previous Year	Straight-line	Projected	Conversion	Projection	Various	Population		Projection	Various	Population	Projection
	1973	Factor	Estimate	to Yearly Impact	for 1974	Adjustments	FY74	FY75	for 1975	Adjustments	FY76	for 1976
Active Duty Dependents	297	2.047	608	12	620	(34)	902,969	908,609	590	(7)	896,762	575
Retired Dependents	427	2.354	1,005	20	1,025	62	824,250	870,088	1,148	1	909,335	1,200
Retired Personnel	110	2.436	268	6	274	35	311,754	329,277	326	0	344,147	341

(psychotherapy)	Previous Year	Straight-line	Projected	Conversion	Projection	Various	Population		Projection	Various	Population	Projection
	1973	Factor	Estimate	to Yearly Impact	for 1974	Adjustments	FY74	FY75	for 1975	Adjustments	FY76	for 1976
Active Duty Dependents	174	4.109	715	0	715	(339)	902,969	908,609	378	0	896,762	373
Retired Dependents	124	3.411	423	0	423	(194)	824,250	870,088	242	0	909,335	253
Retired Personnel	24	3.917	94	0	94	(1)	311,754	329,277	98	(1)	344,147	102

Cost Per Day (CPD)

Inpatient Care

	Previous Year	Change in	Projection	Predicted	Projection	Predicted	Projection
	1973	1973	for 1974	Inflation '74	for 1975	Inflation '75	for 1976
Active Duty Dependents	$122 70	10 90%	$136 07	15.00%	$156 48	0.00%	$156 48
Retired Dependents	$67 73	16 90%	$79 18	15.00%	$91 06	0.00%	$91 06
Retired Personnel	$77.39	27.00%	$98 29	15.00%	$113 03	0.00%	$113 03

Outpatient Care

(other than psychotherapy)	Projection	Predicted	Projection	Predicted	Projection
	for 1974	Inflation '75	for 1975	Inflation '76	for 1976
Active Duty Dependents	$18 13	15.00%	$20 85	0.00%	$20 85
Retired Dependents	$15 86	15.00%	$18 24	0.00%	$18 24
Retired Personnel	$19 99	15.00%	$22 99	0.00%	$22 99

(psychotherapy)	Projection	Predicted	Projection	Predicted	Projection
	for 1974	Inflation '75	for 1975	Inflation '76	for 1976
Active Duty Dependents	$23 80	15.00%	$27 37	0.00%	$27 37
Retired Dependents	$21 69	15.00%	$24 94	0.00%	$24 94
Retired Personnel	$22 34	15.00%	$25 69	0.00%	$25 69

Illustration 6.2: Derivation Values for CHAMPUS Budget

forecasting techniques were employed and the projection was held constant during the three years that were estimated in the budget. Inflation is also based upon a one year trend. The inflation that was experienced during 1973 was calculated and projected to be the same during the next two years with no inflation in the last year. The methods from which population was calculated were not provided by the reference.

The Monte Carlo financial budgeting scenario was performed by assigning probabilistic distribution to the forty-eight variables with total obligations as the objective. Since neither historic data nor CHAMPUS manager experience was available, there was no accurate method to determine appropriate distributions. Instead, reasonable variations around the actual values were used with uniform, triangular, and tabular distributions. This method demonstrates how a spreadsheet and Monte Carlo method can be applied for financial budgeting which is within the scope of this thesis. There was no attempt made to design a better CHAMPUS financial budgeting system which would require an anlaysis of sufficient historical data.

Figure 6.2 presents the frequency distribution for 500 and 750 iterations. The distribution appears reasonably stable thus sufficient iterations were performed. The distributions do not reflect any resemblance to factual data since the inputs were only theorized. However, Figure 6.2 does show the benefit of Monte Carlo simulation. Instead of the deterministic values of $527,383 as provided in Illustration 6.1, a range of values is depicted. For example, there is approximately a 80% chance that total obligation will exceed $580,000 while only a 20% chance that it will exceed $780,000. Furthermore, even though the inputted values are not factual but only a variation of the deterministic values, the probability of achieving a budget of $527,383 is less than 10% in this scenario. This may explain why CHAMPUS consistently exceeds budget every year. The benefit of

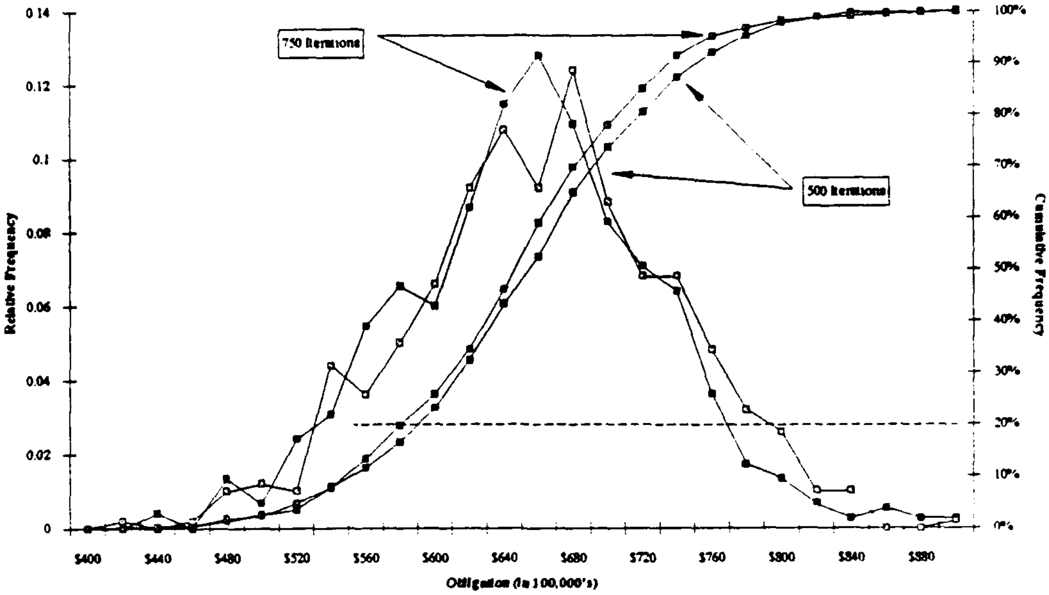


Figure 6.2: Total Simulated Obligations

this information is enormous. A manager does not have to base his/her judgment solely upon a single value but instead can weigh the *probability of achieving* a goal based upon a frequency distribution.

This scenario illustrates the benefit of Monte Carlo simulation in financial budgeting. As stated previously, this macro can also be applied in many other resource allocation scenarios. Iι is applicable to situations that involve uncertainty that can be reasonably quantified such as capital investment scenarios. The next chapter presents the summary and conclusions for this simulation method as well as the previous two chapters' simulations.

VII. SUMMARY AND CONCLUSIONS

The overall purpose of this study was to explore the possibility of applying digital computer spreadsheets as a sophisticated tool for resolving resource allocation enigmas. This involved a general discussion of simulation as a resource allocation tool and the methodology required to adapt a spreadsheet as a viable simulation device. To illustrate spreadsheet simulation, three separate and unique resource allocation scenarios were analyzed.

An inventory distribution system was the first allocation dilemma that was simulated. Several spreadsheet macros were created to fully analyze the behavior of the complex system. *Microsoft's Excel 4.0* was fully capable of the simulation task and many techniques that were created were used in the next two scenarios.

The second scenario was a queuing utilization analysis of a hospital emergency room. This model proved to be the most difficult to adapt to a spreadsheet due to the requirement of maintaining a vast data base on present and past events. However, through creative programming techniques, the spreadsheet was also capable of queuing simulation.

Financial budgeting through Monte Carlo methodology was the final simulation scenario analyzed. Adapting the spreadsheet for this scenario was the least difficult of the three. The result of Monte Carlo simulation is not true simulation as the produced result is a probability distribution vice a system behavior analysis. However, the Monte Carlo is the most adaptable of the three models as it is not as specific in design as the previous two models. It can be applied to a wide variety of resource allocation models.

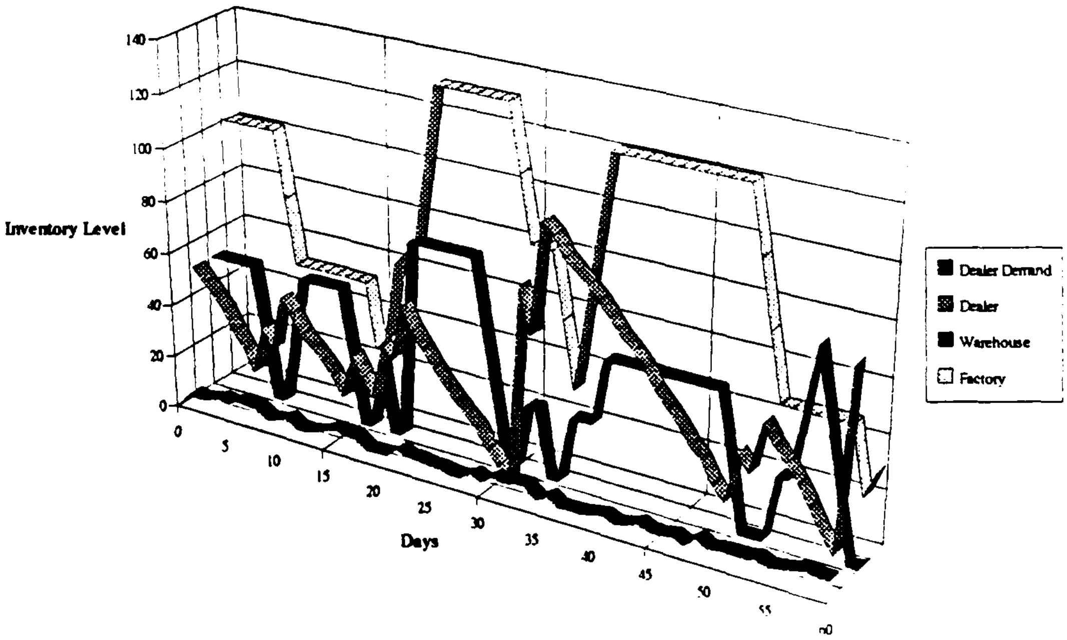
Simulation methodology for resource allocation is no longer limited to those who have access to simulation specific computer software. Spreadsheets, that are available to virtually every manager, can be programmed in simulation methodology to anaiyze all but the most complex resource allocation enigma.

APPENDIX A
INVENTORY DISTRIBUTION OUTPUT AND MACROS

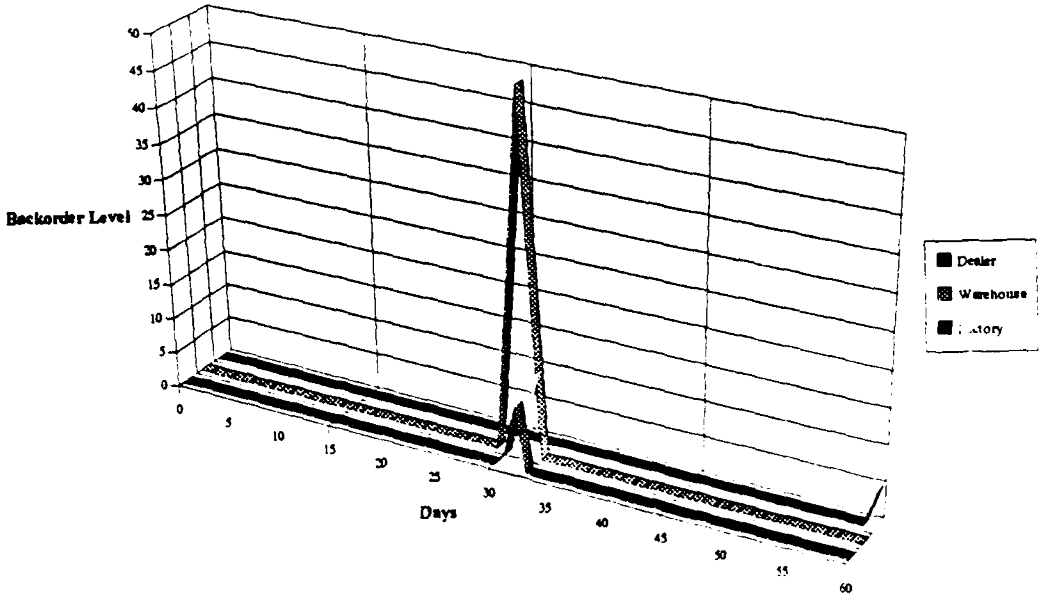
SCENARIO ONE—ONE WAREHOUSE, NO USER INVOLVEMENT

Initial Data	Dealer	Warehouse	Dealer
Beginning Inventory	50	50	100
Reorderr/Begin Production Level	**25**	**25**	**35**
Stop Production	*	*	100
Amount Order/Rate of Production	**25**	**25**	**30**
Holding Cost	$0.10	$0.10	$0.10
Order/Setup Cost	**$20**	**$20**	**$100**
Shortage Cost	$50	$50	$50

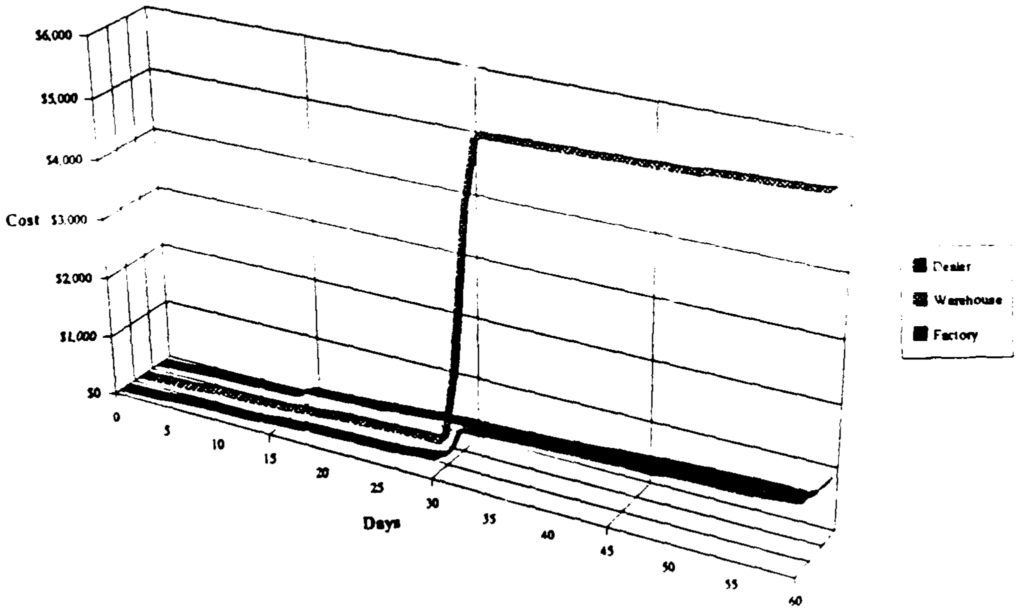
Inventory



Backorders



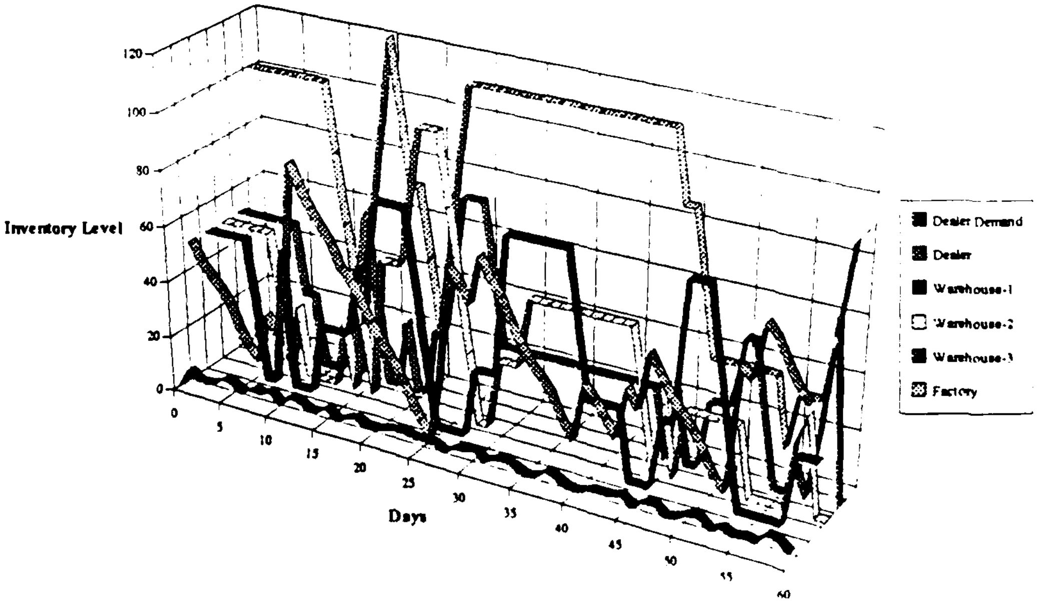
Total Cost



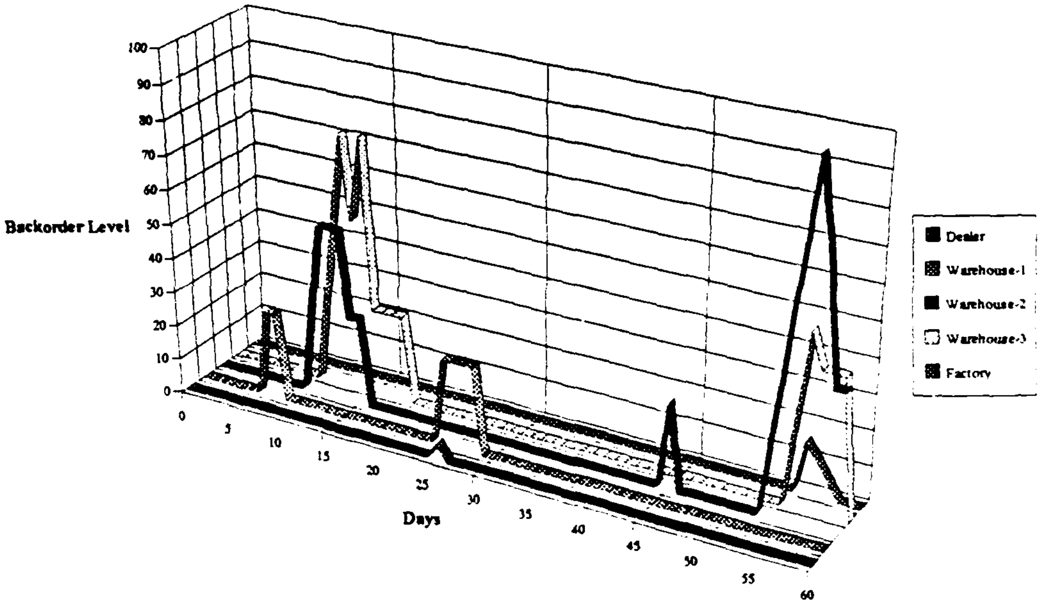
SCENARIO 2—THREE WAREHOUSES, NO USER INVOLVEMENT

Initial Data	Dealer	Warehouse	Dealer
Beginning Inventory	50	50	100
Reorder/Begin Production Level	25	25	35
Stop Production	*	*	100
Amount Order/Rate of Production	25	25	30
Holding Cost	$0.10	$0.10	$0.10
Order/Setup Cost	$20	$20	$100
Shortage Cost	$50	$50	$50

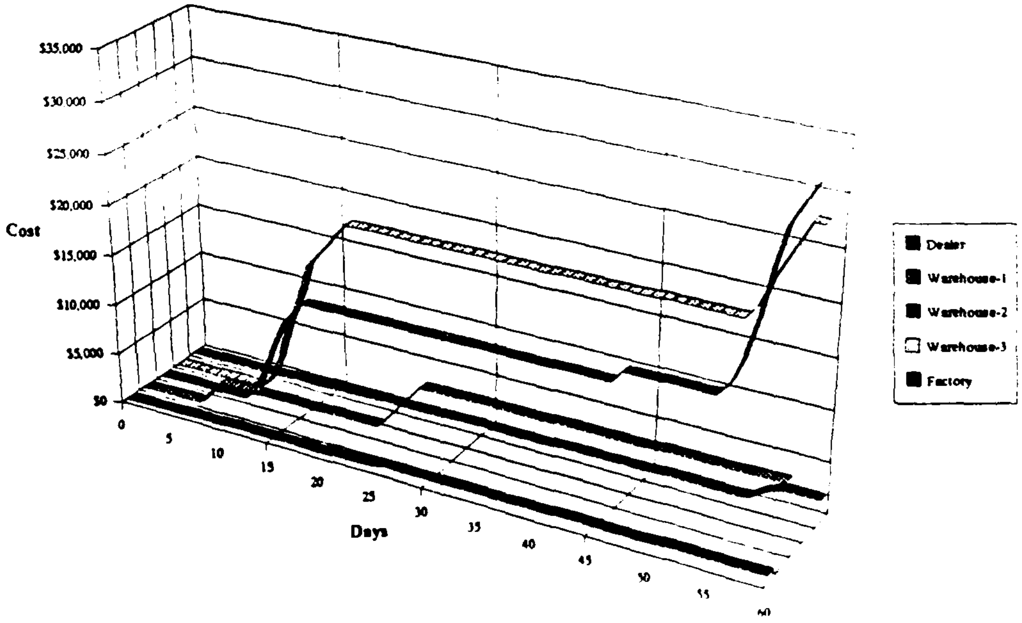
Inventory



Backorder



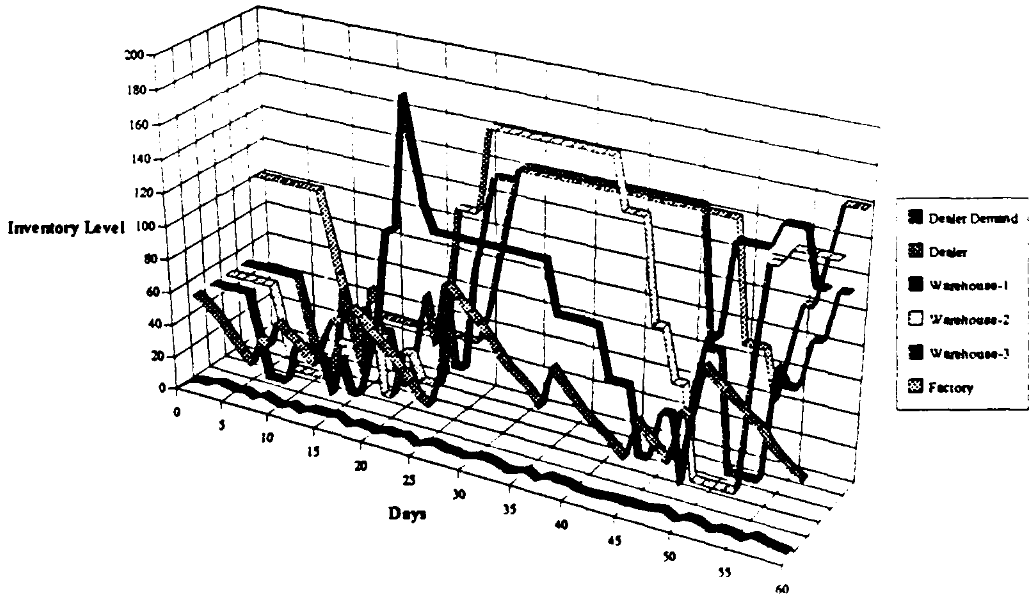
Total Cost



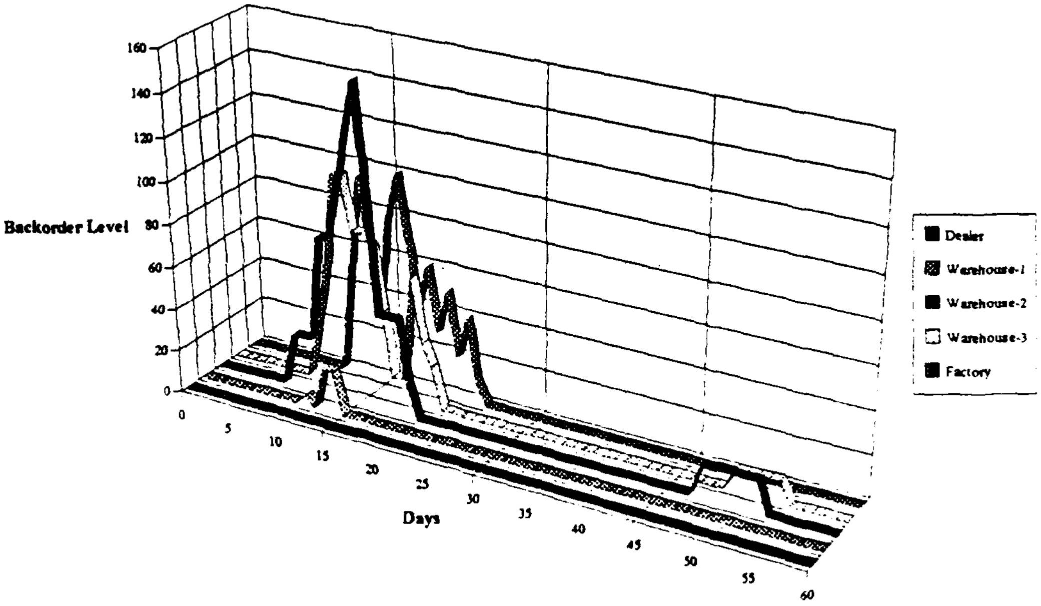
SCENARIO 3—THREE WAREHOUSES, FULL USER INVOLVEMENT

Initial Data	Dealer	Warehouse	Dealer
Beginning Inventory	50	50	100
Reorder/Begin Production Level	25	25	35
Stop Production	*	*	100
Amount Order/Rate of Production	25	25	30
Holding Cost	$0.10	$0.10	$0.10
Order/Setup Cost	$20	$20	$100
Shortage Cost	$50	$50	$50

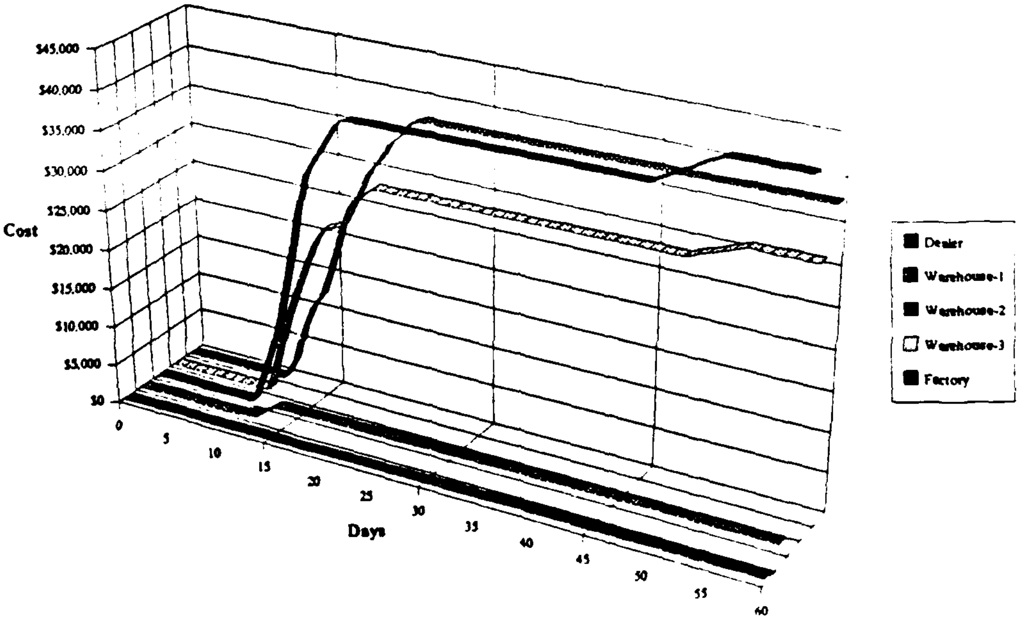
Inventory



Backorder



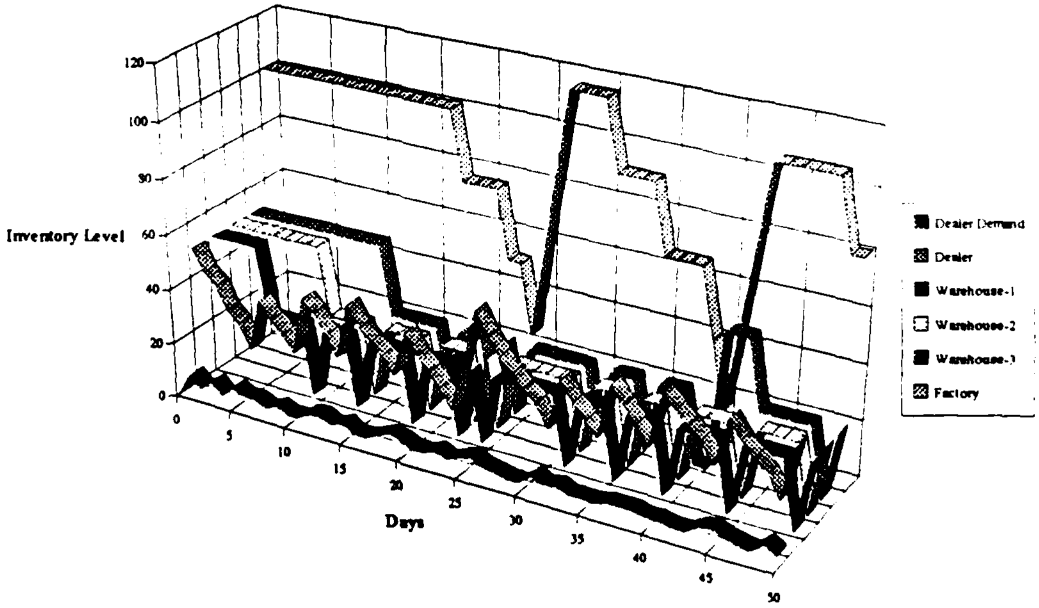
Total Cost



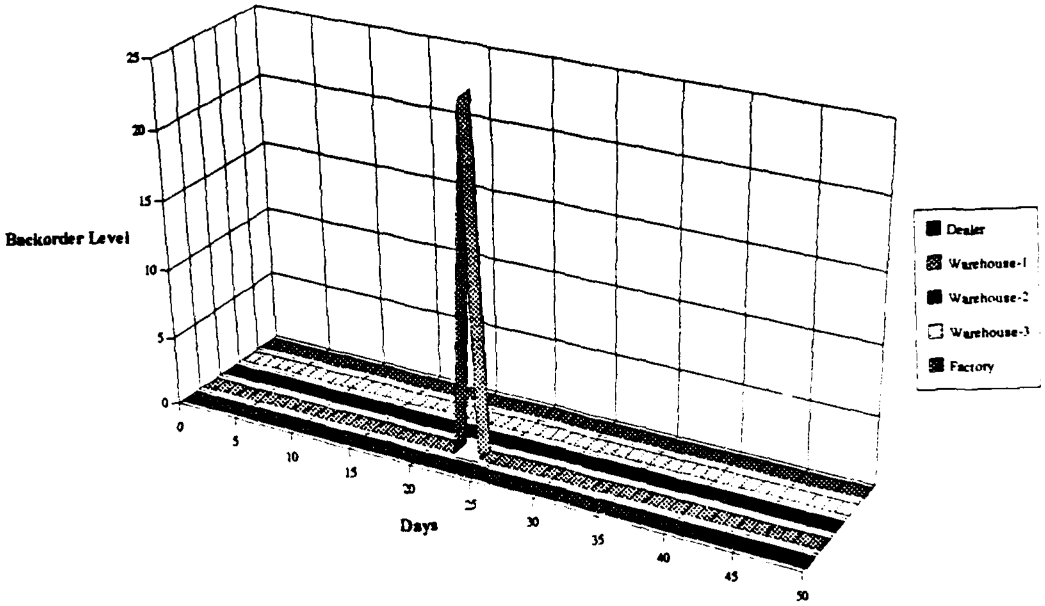
SCENARIO 4—THREE WAREHOUSES, JIT ENVIRONMENT

Initial Data	Dealer	Warehouse	Dealer
Beginning Inventory	50	50	100
Reorder/Begin Production Level	25	25	35
Stop Production	*	*	100
Amount Order/Rate of Production	25	25	30
Holding Cost	$0.10	$0.10	$0.10
Order/Setup Cost	$20	$20	$100
Shortage Cost	$50	$50	$50

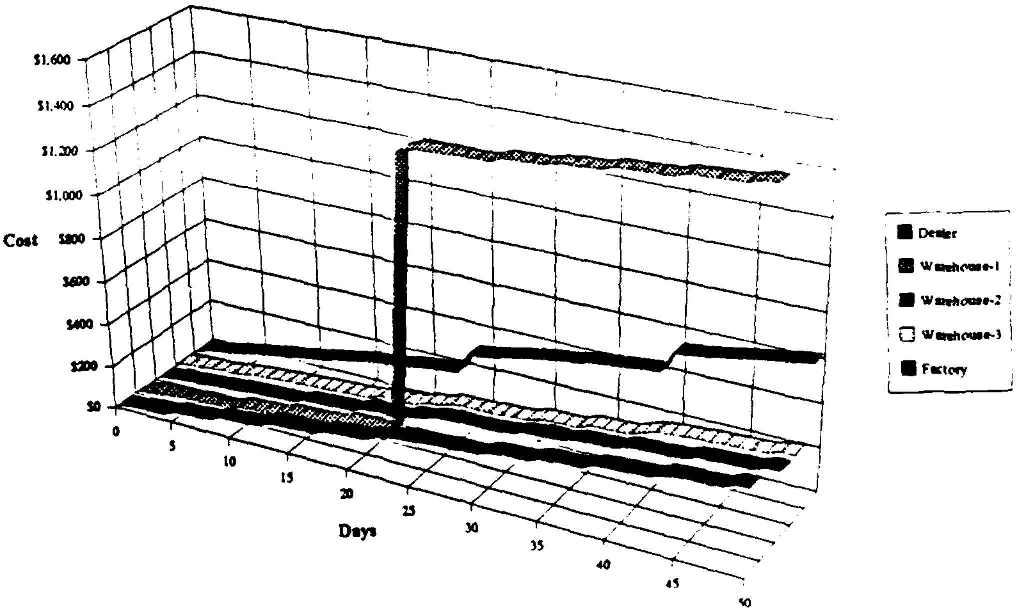
Inventory



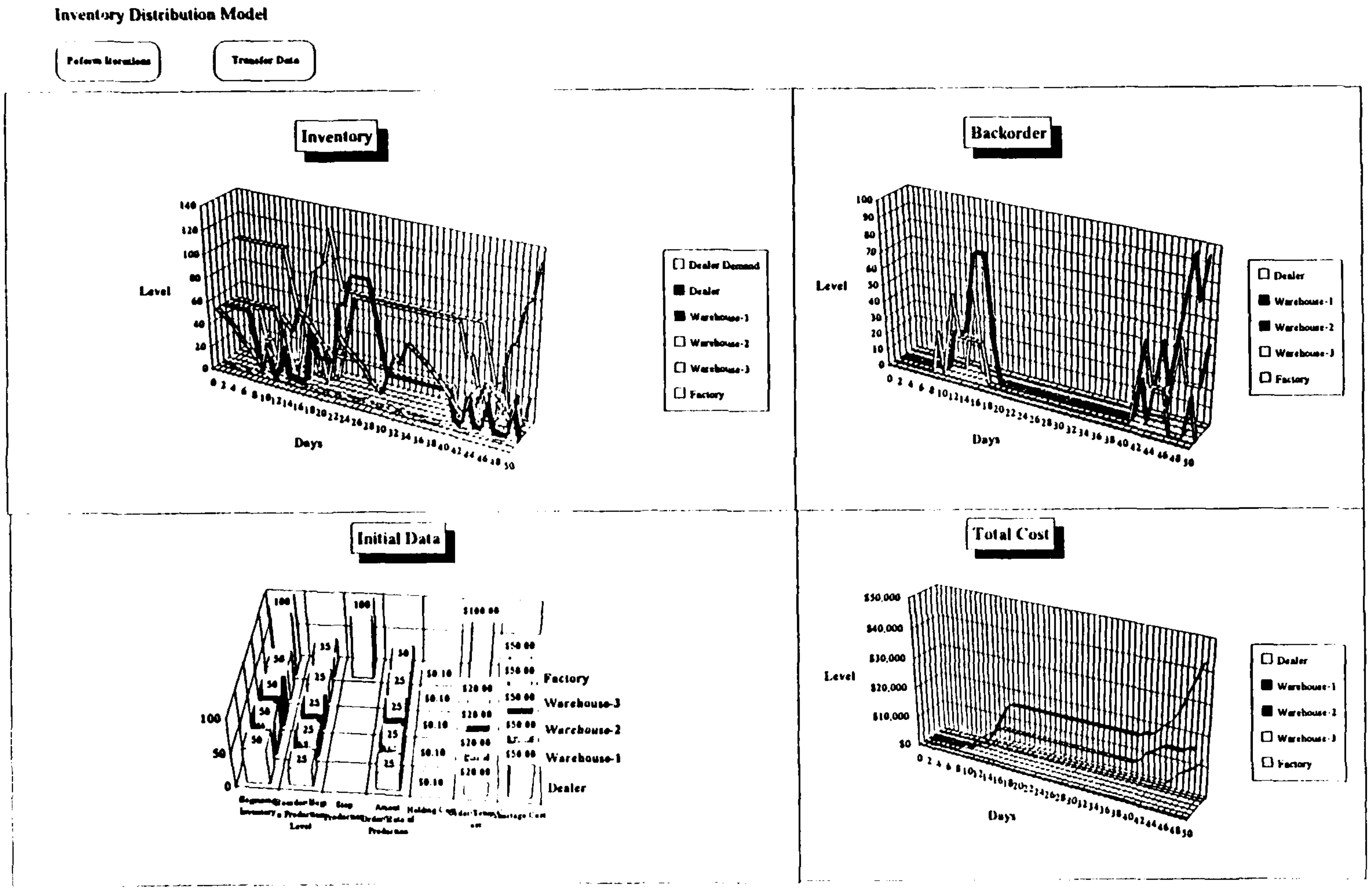
Backorders



Total Cost



INVENTORY DISTRIBUTION WORKSHEET



Inventory Distribution Model

MASTER DISTRIBUTION MANAGEMENT MACRO

	A	B	C
1	*names*	Commands	*comments*
2			
3		Summary Information	
4	*Title:*	*Master Distribution Management Macro*	
5	*Version:*	v1.0	
6	*Author*	Dixon Hicks	
7	*Corporation:*	Naval Postgraduate School	
8	*Creation Date:*	Wednesday, January 20, 1993	
9			
10	*CommandWindow*	CommandWindow	
11	[illegible]	=WINDOW.MAXIMIZE("MASTER.XLS")	Maximize Master Macro
12		=RETURN()	
13			
14	Master_Close	=ECHO(FALSE)	Close Worksheets and Macros
15		=ACTIVATE("FACTORY.XLM")	
16		=CLOSE(FALSE)	
17		=ACTIVATE("WARE-1.XLM")	
18		=CLOSE(FALSE)	
19		=ACTIVATE("WARE-2.XLM")	
20		=CLOSE(FALSE)	
21		=ACTIVATE("WARE-3.XLM")	
22		=CLOSE(FALSE)	
23		=ACTIVATE("DEALER.XLM")	
24		=CLOSE(FALSE)	
25		=ACTIVATE("MASTER.XLM")	
26		=CLOSE(FALSE)	
27		=RETURN()	
28			
29			
30	Master	=ERROR.TYPE(WINDOWS(1,"DEALER.XLM"))	Open Distributor Macros
31		=ERROR.TYPE(WINDOWS(1,"WARE-1.XLM"))	
32		=ERROR.TYPE(WINDOWS(1,"WARE-2.XLM"))	
33		=ERROR.TYPE(WINDOWS(1,"WARE-3.XLM"))	
34		=ERROR.TYPE(WINDOWS(1,"FACTORY.XLM"))	
35		=IF(B30=7,OPEN("DEALER.XLM",3,TRUE))	
36		=IF(B31=7,OPEN("WARE-1.XLM",3,TRUE))	
37		=IF(B32=7,OPEN("WARE-2.XLM",3,TRUE))	
38		=IF(B33=7,OPEN("WARE-3.XLM",3,TRUE))	
39		=IF(B34=7,OPEN("FACTORY.XLM",3,TRUE))	
40		=ACTIVATE("MASTER.XLS")	
41			
42		=SET.VALUE(Prompt_Iteration,FALSE)	*Set Initial Variable*
43		=SET.VALUE(P_Start_Iteration,FALSE)	
44		=SET.VALUE(Screen_Update,TRUE)	
45		=SET.VALUE(Iterations,50)	
46		=SET.VALUE(Startup_Iterations,0)	
47		=SET.VALUE(S_Count,1)	
48	First_Iteration	=SET.VALUE(First_Iteration,TRUE)	
49			
50		=DIALOG.BOX(Input_Box_1)	User Choose Variables
51		=IF(Warehouse_Number>3)	
52		= ALERT("More than 3 Warehouse Selected",3)	
53		= GOTO(B50)	
54		=END.IF()	
55			
56		=IF(B50=FALSE)	End Simulation by user
57	QUIT	= ALERT("Simulation Cancelled",3)	
58		= ACTIVATE("MASTER.XLS")	
59		= HALT()	
60		=END.IF()	
61			
62		=IF(Change_Dealer_Demand=TRUE)	
63		= DIALOG.BOX(Input_Box_2)	
64		=END.IF()	
65			
66		=ECHO(Screen_Update)	Screen Updating
67			
68	Total_Iterations	=Iterations+Startup_Iterations	
69		=SET.NAME("Output_Data",$N$4:OFFSET($N$4,(Total_Iterations+1),31))	Clear Output Data
70		=FORMULA.GOTO(Output_Data,$N$4)	
71		=CLEAR(3)	
72		=VSCROLL(1,TRUE)	
73			
74			
75			
76			**Iteration Routine**
77			
78		=IF(P_Start_Iteration=TRUE)	Perform Startup Iteration if Desired
79		= FOR("Counter_S",1,Startup_Iterations)	
80	S_Count	= Counter_S	
81		= IF(Counter_S>1,SET.VALUE(First_Iteration,FALSE))	
82		= SET.VALUE(Day,Counter_S)	
83		= MESSAGE(TRUE,"Running Startup Iteration")	
84		= GOTO(B97)	
85		= END.IF()	
86			
87		=FOR("Counter",S_Count,(Total_Iterations))	Perform Iterations
88	Day	= Counter	
89			
90		= IF(Day>1,SET.VALUE(First_Iteration,FALSE))	Flag for other Macros

	A	B	C
1	names	Commands	comments
91		= IF(Prompt_Iteration=TRUE)	
92		= ALERT("Perform Another iteration?",1)	Perform another iteration?
93		= IF(B92=FALSE,GOTO(B110),)	
94		= END.IF()	
95			
96		= MESSAGE(TRUE,Day)	Display Day in Message Bar
97		= RUN(DEALER.XLM!Dealer,FALSE)	Call Dealer Macro
98			
99		= IF(Day=1)	
100		= ACTIVATE("MASTER.XLM")	Return Display Control to Master
101		= SET.VALUE(Day,0)	
102		= RUN(MASTER.XLM!Master_Output)	Copy Initial Data
103		= IF(P_Start_Iteration=TRUE,SET.VALUE(Day,Counter_5),SET.VALUE(Day,Counter))	
104		= END.IF()	
105			
106		= RUN(MASTER.XLM!Master_Output)	Copy Each Days Results
107		= IF(P_Start_Iteration=TRUE,GOTO(B115),)	
108		=NEXT()	
109			
110		=ALERT("Simulation Complete",3)	Display Simulation Complete
111		=MESSAGE(FALSE,)	
112		=ACTIVATE("MASTER.XLS")	Return Control To Worksheet
113		=RETURN()	
114			
115		=NEXT()	
116		=SET.VALUE(P_Start_Iteration,FALSE)	Clear Startup Flag
117		=GOTO(B87)	
118			
119			**Output Routine**
120			
121	Master_Output	= FORMULA(Day,OFFSET($N$4,Day,0))	Day
122		= COPY(OFFSET(DEALER.XLM!$T$4,Day,7),OFFSET($N$4,Day,1))	Dealer Demand
123		= COPY(OFFSET(DEALER.XLM!$T$4,Day,1),OFFSET($N$4,Day,2))	Inventory Dealer
124		= COPY(OFFSET('WARE-1.XLM'!$T$4,Day,1),OFFSET($N$4,Day,3))	Warehouse-1
125		= COPY(OFFSET('WARE-2.XLM'!$T$4,Day,1),OFFSET($N$4,Day,4))	Warehouse-2
126		= COPY(OFFSET('WARE-3.XLM'!$T$4,Day,1),OFFSET($N$4,Day,5))	Warehouse-3
127		= COPY(OFFSET(FACTORY.XLM!$N$4,Day,1),OFFSET($N$4,Day,6))	Factory
128		= COPY(OFFSET(DEALER.XLM!$T$4,Day,2),OFFSET($N$4,Day,7))	Backorder Dealer
129		= COPY(OFFSET('WARE-1.XLM'!$T$4,Day,2),OFFSET($N$4,Day,8))	Warehouse-1
130		= COPY(OFFSET('WARE-2.XLM'!$T$4,Day,2),OFFSET($N$4,Day,9))	Warehouse-2
131		= COPY(OFFSET('WARE-3.XLM'!$T$4,Day,2),OFFSET($N$4,Day,10))	Warehouse-3
132		= COPY(OFFSET(FACTORY.XLM!$N$4,Day,2),OFFSET($N$4,Day,11))	Factory
133		= COPY(OFFSET(DEALER.XLM!$T$4,Day,3),OFFSET($N$4,Day,12))	Shortage Cost Dealer
134		= COPY(OFFSET('WARE-1.XLM'!$T$4,Day,3),OFFSET($N$4,Day,13))	Warehouse-1
135		= COPY(OFFSET('WARE-2.XLM'!$T$4,Day,3),OFFSET($N$4,Day,14))	Warehouse-2
136		= COPY(OFFSET('WARE-3.XLM'!$T$4,Day,3),OFFSET($N$4,Day,15))	Warehouse-3
137		= COPY(OFFSET(FACTORY.XLM!$N$4,Day,3),OFFSET($N$4,Day,16))	Factory
138		= COPY(OFFSET(DEALER.XLM!$T$4,Day,4),OFFSET($N$4,Day,17))	Holding Cost Dealer
139		= COPY(OFFSET('WARE-1.XLM'!$T$4,Day,4),OFFSET($N$4,Day,18))	Warehouse-1
140		= COPY(OFFSET('WARE-2.XLM'!$T$4,Day,4),OFFSET($N$4,Day,19))	Warehouse-2
141		= COPY(OFFSET('WARE-3.XLM'!$T$4,Day,4),OFFSET($N$4,Day,20))	Warehouse-3
142		= COPY(OFFSET(FACTORY.XLM!$N$4,Day,4),OFFSET($N$4,Day,21))	Factory
143		= COPY(OFFSET(DEALER.XLM!$T$4,Day,5),OFFSET($N$4,Day,22))	Ordering Cost Dealer
144		= COPY(OFFSET('WARE-1.XLM'!$T$4,Day,5),OFFSET($N$4,Day,23))	Warehouse-1
145		= COPY(OFFSET('WARE-2.XLM'!$T$4,Day,5),OFFSET($N$4,Day,24))	Warehouse-2
146		= COPY(OFFSET('WARE-3.XLM'!$T$4,Day,5),OFFSET($N$4,Day,25))	Warehouse-3
147		= COPY(OFFSET(FACTORY.XLM!$N$4,Day,5),OFFSET($N$4,Day,26))	Setup Cost Factory
148		= COPY(OFFSET(DEALER.XLM!$T$4,Day,6),OFFSET($N$4,Day,27))	Total Cost Dealer
149		= COPY(OFFSET('WARE-1.XLM'!$T$4,Day,6),OFFSET($N$4,Day,28))	Warehouse-1
150		= COPY(OFFSET('WARE-2.XLM'!$T$4,Day,6),OFFSET($N$4,Day,29))	Warehouse-2
151		= COPY(OFFSET('WARE-3.XLM'!$T$4,Day,6),OFFSET($N$4,Day,30))	Warehouse-3
152		= COPY(OFFSET(FACTORY.XLM!$N$4,Day,6),OFFSET($N$4,Day,31))	Factory
153			
154		=RETURN()	Add other parameters if Desired
155			
156			
157			***Data Transfer Routine***
158			
159			
160	Master_Data	=ECHO(FALSE)	Transfer Data to Worksheet
161			
162		=COPY(Output_Data,'MASTER-T.XLS'!$AQ$4:OFFSET('MASTER-T.XLS'!$AQ$4,(Iterations+Startup_Iterations+1),31))	
163			
164		=COPY(DEALER.XLM!Beginning_Inventory,'MASTER-T.XLS'!BY3)	Transfer Initial Values to Worksheet
165		=COPY(DEALER.XLM!Reorder_Point,'MASTER-T.XLS'!BY4)	
166		=COPY(DEALER.XLM!Reorder_Quantity,'MASTER-T.XLS'!BY6)	
167		=COPY(DEALER.XLM!Holding_Cost,'MASTER-T.XLS'!BY7)	
168		=COPY(DEALER.XLM!Cost_Order,'MASTER-T.XLS'!BY8)	
169		=COPY(DEALER.XLM!Shortage_Cost,'MASTER-T.XLS'!BY9)	
170		=COPY('WARE-1.XLM'!Beginning_Inventory,'MASTER-T.XLS'!BZ3)	
171		=COPY('WARE-1.XLM'!Reorder_Point,'MASTER-T.XLS'!BZ4)	
172		=COPY('WARE-1.XLM'!Reorder_Quantity,'MASTER-T.XLS'!BZ6)	
173		=COPY('WARE-1.XLM'!Holding_Cost,'MASTER-T.XLS'!BZ7)	
174		=COPY('WARE-1.XLM'!Cost_Order,'MASTER-T.XLS'!BZ8)	
175		=COPY('WARE-1.XLM'!Shortage_Cost,'MASTER-T.XLS'!BZ9)	
176		=COPY('WARE-2.XLM'!Beginning_Inventory,'MASTER-T.XLS'!CA3)	
177		=COPY('WARE-2.XLM'!Reorder_Point,'MASTER-T.XLS'!CA4)	
178		=COPY('WARE-2.XLM'!Reorder_Quantity,'MASTER-T.XLS'!CA6)	
179		=COPY('WARE-2.XLM'!Holding_Cost,'MASTER-T.XLS'!CA7)	
180		=COPY('WARE-2.XLM'!Cost_Order,'MASTER-T.XLS'!CA8)	
181		=COPY('WARE-2.XLM'!Shortage_Cost,'MASTER-T.XLS'!CA9)	
182		=COPY('WARE-3.XLM'!Beginning_Inventory,'MASTER-T.XLS'!CB3)	
183		=COPY('WARE-3.XLM'!Reorder_Point,'MASTER-T.XLS'!CB4)	
184		=COPY('WARE-3.XLM'!Reorder_Quantity,'MASTER-T.XLS'!CB6)	
185		=COPY('WARE-3.XLM'!Holding_Cost,'MASTER-T.XLS'!CB7)	

	A	3	C
1	names	Commands	comments
186		=COPY("WARE-3 XLM'!Cost_Order,"MASTER-T XLS"!CB8)	
187		=COPY("WARE-3 XLM'!Shortage_Cost,"MASTER-T XLS"!CB9)	
188		=COPY(FACTORY XLM'Beginning_Inventory,"MASTER-T XLS"!CC3)	
189		=COPY(FACTORY XLM'Begin_Production_Level,"MASTER-T XLS"!CC4)	
190		=COPY(FACTORY XLM'Stop_Production_Level,"MASTER-T XLS"!CC5)	
191		=COPY(FACTORY XLM'Production_Rate,"MASTER-T XLS"!CC6)	
192		=COPY(FACTORY XLM'Holding_Cost,"MASTER-T XLS"!CC7)	
193		=COPY(FACTORY XLM'Setup_Cost,"MASTER-T XLS"!CC8)	
194		=COPY(FACTORY XLM'Shortage_Cost,"MASTER-T XLS"!CC9)	
195			
196		=DEFINE.NAME("Master_Day","MASTER-T.XLS"!$AQ$4:OFFSET("MASTER-T XLS"!$AQ$4,MASTER.XLM!Total_Iterations	Define Names for Graphs
197		=DEFINE.NAME("Dealer_Demand","MASTER-T XLS"!$AR$4:OFFSET("MASTER-T XLS"!$AR$4,MASTER XLM!Total_Itera	
198		=DEFINE.NAME("Dealer_I","MASTER-T.XLS"!$AS$4:OFFSET("MASTER-T.XLS"!$AS$4,MASTER.XLM!Total_Iterations,0))	
199		=DEFINE.NAME("Warehouse_1_I","MASTER-T.XLS"!$AT$4:OFFSET("MASTER-T.XLS"!$AT$4,MASTER.XLM!Total_Iterat	
200		=DEFINE.NAME("Warehouse_2_I","MASTER-T.XLS"!$AU$4:OFFSET("MASTER-T.XLS"!$AU$4,MASTER.XLM!Total_Itera	
201		=DEFINE.NAME("Warehouse_3_I","MASTER-T.XLS"!$AV$4:OFFSET("MASTER-T.XLS"!$AV$4,MASTER.XLM!Total_Itera	
202		=DEFINE.NAME("Factory_I","MASTER-T.XLS"!$AW$4:OFFSET("MASTER-T.XLS"!$AW$4,MASTER.XLM!Total_Iterations,	
203		=DEFINE.NAME("Dealer_BO","MASTER-T.XLS"!$AX$4:OFFSET("MASTER-T.XLS"!$AX$4,MASTER.XLM!Total_Iterations,	
204		=DEFINE.NAME("Warehouse_1_BO","MASTER-T.XLS"!$AY$4:OFFSET("MASTER-T.XLS"!$AY$4,MASTER.XLM!Total_Ite	
205		=DEFINE.NAME("Warehouse_2_BO","MASTER-T.XLS"!$AZ$4:OFFSET("MASTER-T.XLS"!$AZ$4,MASTER.XLM!Total_Iter	
206		=DEFINE.NAME("Warehouse_3_BO","MASTER-T.XLS"!$BA$4:OFFSET("MASTER-T.XLS"!$BA$4,MASTER.XLM!Total_Ite	
207		=DEFINE.NAME("Factory_BO","MASTER-T.XLS"!$BB$4:OFFSET("MASTER-T.XLS"!$BB$4,MASTER.XLM!Total_Iterations	
208		=DEFINE.NAME("Dealer_Cost","MASTER-T.XLS"!$BR$4:OFFSET("MASTER-T.XLS"!$BR$4,MASTER.XLM!Total_Iteration	
209		=DEFINE.NAME("Warehouse_1_Cost","MASTER-T XLS"!$BS$4:OFFSET("MASTER-T XLS"!$BS$4,MASTER.XLM!Total_Ite	
210		=DEFINE.NAME("Warehouse_2_Cost","MASTER-T.XLS"!$BT$4:OFFSET("MASTER-T.XLS"!$BT$4,MASTER.XLM!Total_It	
211		=DEFINE.NAME("Warehouse_3_Cost","MASTER-T XLS"!$BU$4:OFFSET("MASTER-T.XLS"!$BU$4,MASTER.XLM!Total_It	
212		=DEFINE.NAME("Factory_Cost","MASTER-T XLS"!$BV$4:OFFSET("MASTER-T XLS"!$BV$4,MASTER.XLM!Total_Iteratio	
213			
214		=ZOOM(60)	
215		=ACTIVATE("MASTER.XLS")	
216			
217		=RETURN()	

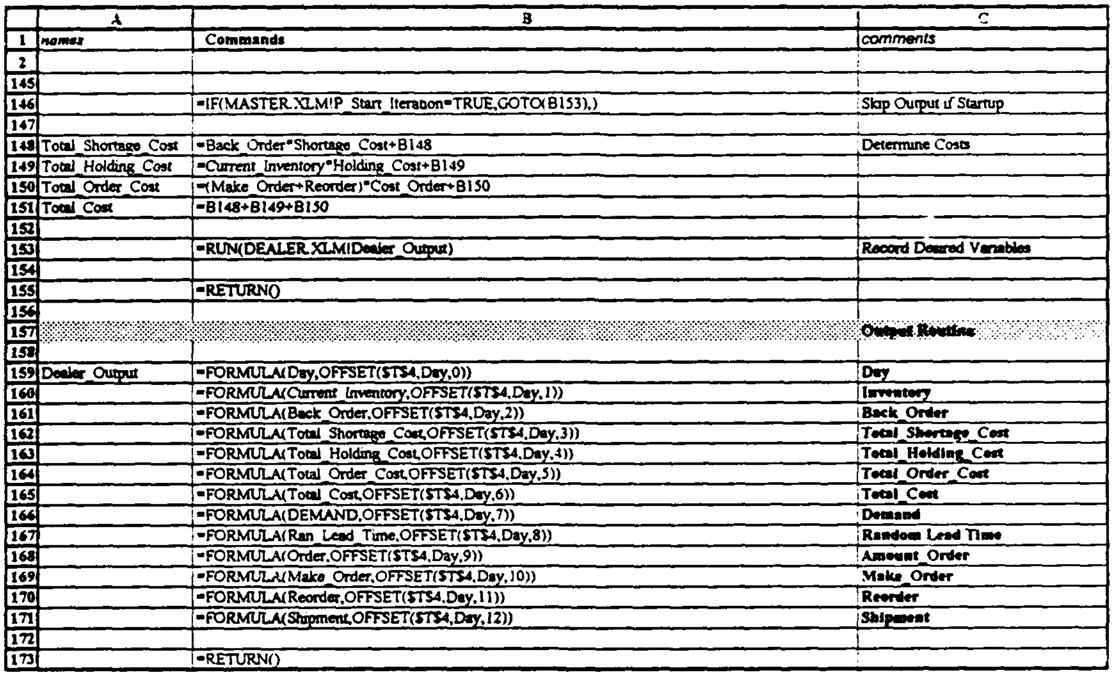
	E	F	G	H	I	J	K	L
1	DIALOGS							
23	Input_Box_1							
24		121	73	926	335	Initial Iteration and Policy Variables Da		
25	3	816	21	88		DONE		
26	13	18	30			Ask for Individual Data During Iteratio	FALSE	Prompt_Individual
27	13	18	52			Change Daily Dealer Demand	FALSE	Change_Dealer_Demand
28	13	18	7			Allow Screen Updating (Slows Simulati	FALSE	Screen_Update
29	5	43	80			Number of Warehouses (MAX 3)		
30	7	18	77	17			3	Warehouse_Number
31	14	410	11	348	96	Iterations During Simulation		
32	13					Prompt for Each Time Iteration	FALSE	Prompt_Iteration
33	13					Run a Startup Iteration	FALSE	P_Start_Iteration
34	5	442	61			Number of Iterations		
35	7	684	54	50			5	Iterations
36	5	442	80			Number of Startup Iterations		
37	7	684	78	50			0	Startup_Iterations
38	14	18	151	279	169	Dealer Policy Variables		
39	5	28	173			Beginning Inventory		
40	7	200	171	75			50	Dealer_BI
41	5	28	198			Reorder Level		
42	7	200	195	75			25	Dealer_RL
43	5	28	222			Reorder Quantity		
44	7	200	219	75			25	Dealer_RQ
45	5	28	246			Holding Cost		
46	8	200	243	75			0.1	Dealer_HC
47	5	28	268			Ordering Cost		
48	8	200	266	75			20	Dealer_OC
49	5	28	289			Shortage Cost		
50	8	200	289	75			50	Dealer_SC
51	14	311	152	279	168	Warehouse Policy Variables		
52	5	321	174			Beginning Inventory		
53	7	496	172	75			50	Warehouse_BI
54	5	321	199			Reorder Level		
55	7	496	196	75			25	Warehouse_RL
56	5	321	223			Reorder Quantity		
57	7	496	220	75			25	Warehouse_RQ
58	5	321	247			Holding Cost		
59	8	496	244	75			0.1	Warehouse_HC
60	5	321	269			Ordering Cost		
61	8	496	267	75			20	Warehouse_OC
62	5	320	290			Shortage Cost		
63	8	496	290	75			50	Warehouse_SC
64	14	603	129	301	191	Factory Policy Variables		
65	5	613	149			Beginning Inventory		
66	7	810	145	75			100	Factory_BI
67	5	613	173			Start Production Level		
68	7	810	171	75			35	Factory_Start
69	5	613	198			Stop Production Level		
70	7	810	195	75			100	Factory_Stop
71	5	613	222			Factory Production Rate		
72	7	810	219	75			30	Factory_Rate
73	5	613	246			Holding Cost		
74	8	810	243	75			0.1	Factory_HC
75	5	613	268			Setup Cost		
76	8	810	266	75			100	Factory_PC
77	5	613	289			Shortage Cost		
78	8	810	289	75			50	Factory_SC
79	2	817	51	88		Quit		

	E	F	G	H	I	J	K	L
1	DIALOGS							
81	Input_Box_2							
82		245	121	487	200	Change Dealer Demand		
83	3	382	19	88		DONE		
84	14	22	14	308	171	Daily Dealer Demand		
85	11						5	Daily_Demand
86	12		54		21	0		
87	12					1		
88	12					2		
89	12					3		
90	12					4		
91	5	99	31			Demand		
92	8	104	53	46			4	Demand_0
93	8	106	77	46			5	Demand_1
94	8	106	100	46			6	Demand_2
95	8	106	123	46			7	Demand_3
96	8	105	144	46			8	Demand_4
97	5	201	31			Frequency		
98	5							
99	8	214	52	46			20	Demand_Freq_0
100	8	214	76	46			20	Demand_Freq_1
101	8	215	98	46			20	Demand_Freq_2
102	8	215	121	46			20	Demand_Freq_3
103	8	215	143	46			20	Demand_Freq_4
104	2	382	50	88		Quit		

DEALER MANAGEMENT MACRO

	A	B	C
1	*names*	**Commands**	*comments*
2			
3		**Summary Information**	
4	*Title:*	Warehouse # 3 Management Macro	
5	*Version:*	v1.0	
6	*Author*	Dixon Hicks	
7	*Corporation:*	Naval Postgraduate School	
8	*Creation Date:*	Wednesday, January 20, 1993	
9			
10	*CommandWindow*	**CommandWindow**	
11	Warehouse	=IF(MASTER.XLM!First_Iteration=TRUE)	*Start Warehouse Macro*
12			
13		= SET.VALUE(Beginning_Inventory,MASTER.XLM!Warehouse_BI)	Set Initial Variable
14		= SET.VALUE(Prompt_Reorder,FALSE) .	
15		= SET.VALUE(Prompt_Quantity,FALSE)	
16		= SET.VALUE(Reorder_Point,MASTER.XLM!Warehouse_RL)	
17		= SET.VALUE(Reorder_Quantity,MASTER.XLM!Warehouse_RQ)	
18		= SET.VALUE(Holding_Cost,MASTER.XLM!Warehouse_HC)	
19		= SET.VALUE(Cost_Order,MASTER.XLM!Warehouse_OC)	
20		= SET.VALUE(Shortage_Cost,MASTER.XLM!Warehouse_SC)	
21		= SET.VALUE(Lead_Time,5)	
22		= SET.VALUE(Day,0)	
23		= SET.VALUE(Shipment,0)	
24		= SET.VALUE(Order,0)	
25		= SET.VALUE(Back_Order,0)	
26		= SET.VALUE(Total_Cost,0)	
27		= SET.VALUE(Total_Holding_Cost,0)	
28		= SET.VALUE(Total_Order_Cost,0)	
29		= SET.VALUE(Total_Shortage_Cost,0)	
30		= SET.VALUE(Received_Order,0)	
31		= SET.VALUE(Reorder,FALSE)	
32		= SET.VALUE(Make_Order,FALSE)	
33		= SET.VALUE(B128,0)	
34		= SET.VALUE(B129,0)	
35		= SET.VALUE(B130,0)	
36		= SET.VALUE(B131,0)	
37		= SET.VALUE(B132,0)	
38			
39		= IF(MASTER.XLM!Prompt_Individual=TRUE)	
40		= *DIALOG.BOX(Input_Box)*	User Choose Variables
41		= IF(B40=FALSE)	End Simulation by user
42	QUIT	= ALERT("Simulation Cancelled",3)	
43		= MESSAGE(FALSE)	
44		= ACTIVATE("MASTER.XLS")	
45		= HALT()	
46		= END IF()	
47		= END.IF()	
48			
49		= SET.VALUE(Current_Inventory,Beginning_Inventory)	Set Current Inventory
50		= GOTO(Lead_Time_Table)	Determine Relative Frequency
51			
52		= SET.NAME("Output_Data",$T$4:OFFSET($T$4,(MASTER.XLM!Total_Iterations+1),12))	Clear Output Data
53		= FORMULA.GOTO(Output_Data,T4)	
54		= CLEAR(3)	
55		= VSCROLL(1,TRUE)	
56		= RUN('WARE-3.XLM'!Warehouse_Output)	Record Initial Variables
57		=END.IF()	
58			
59			
60			**Begin Routine**
61			
62	Day	=MASTER.XLM!Day	Counter for Reference
63			
64		=RAND()	Determine Lead Time
65		=IF(B$64>(VLOOKUP(1,Lead_Time_Table,4)),2,1)	
66		=IF(B$64>(VLOOKUP(2,Lead_Time_Table,4)),3,1)	
67		=IF(B$64>(VLOOKUP(3,Lead_Time_Table,4)),4,1)	
68		=IF(B$64>(VLOOKUP(4,Lead_Time_Table,4)),5,1)	
69	Ran_Lead_Time	=MAX(B65:B68)	

	A	3	C
1	*names*	**Commands**	*comments*
2			
72			
73			
74			**Begin Routine**
75			
76	Day	=MASTER.XLM!Day	Counter for Reference
77			
78		=RAND()	Determine Daily Demand
79		=IF(B$78>(VLOOKUP(0,Demand_Table,4)),Demand_1,Demand_0)	
80		=IF(B$78>(VLOOKUP(1,Demand_Table,4)),Demand_2,Demand_0)	
81		=IF(B$78>(VLOOKUP(2,Demand_Table,4)),Demand_3,Demand_0)	
82		=IF(B$78>(VLOOKUP(3,Demand_Table,4)),Demand_4,Demand_0)	
83	DEMAND	=MAX(B79:B82)	
84			
85		=RAND()	Determine Lead Time
86		=IF(B$85>(VLOOKUP(1,Lead_Time_Table,4)),2,1)	
87		=IF(B$85>(VLOOKUP(2,Lead_Time_Table,4)),3,1)	
88		=IF(B$85>(VLOOKUP(3,Lead_Time_Table,4)),4,1)	
89		=IF(B$85>(VLOOKUP(4,Lead_Time_Table,4)),5,1)	
90	Ran_Lead_Time	=MAX(B86:B89)	
91			
92		=Current_Inventory-Back_Order	Fill Backorder if Possible
93		=IF(B92>=0,0,ABS(B92))	Determine Size of Backorder
94		=IF(B92>0,B92,0)	Determine Change in Inventory
95			
96		=B94-DEMAND	Fill Demand if Possible
97		=IF(B96>=0,B96,0)	Zero Inventory if Negative
98	Back_Order	=IF(B96<0,B93+ABS(B96),B93)	Backorder if Inventory Insufficient
99			
100	Current_Inventory	=B97+Shipment	Receive Shipment if Present
101			
102		=IF(MASTER.XLM!P_Start_Iteration=TRUE,GOTO(Reorder),)	Skip Start Reorder ? if Startup
103			
104		=IF(Prompt_Reorder=TRUE)	Reorder & Quantity?
105		= SET.VALUE(K120,Current_Inventory-Back_Order)	
106		= SET.VALUE(K116,Day)	
107		= SET.VALUE(K118,DEMAND)	
108		= SET.VALUE(Make_Order,FALSE)	
109		= DIALOG.BOX(Reorder_Box)	
110		= IF(B109=FALSE,SET.VALUE(Prompt_Reorder,FALSE),)	
111		= GOTO(Order)	Goto make Order
112		=END.IF()	
113			
114	Reorder	=IF(Current_Inventory<Reorder_Point,TRUE,FALSE)	Make Order
115			
116		=IF(MASTER.XLM!P_Start_Iteration=TRUE,GOTO(Order),)	Skip Reorder Quantity ? if Startup
117			
118		=IF(Reorder=TRUE)	Quantity?
119		= IF(Prompt_Quantity=TRUE)	
120		= SET.VALUE(K130,Day)	
121		= SET.VALUE(K132,DEMAND)	
122		= SET.VALUE(K125,Reorder_Quantity)	
123		= SET.VALUE(K134,Current_Inventory-Back_Order)	
124		= DIALOG.BOX(Order_Box)	
125		= IF(B124=FALSE,SET.VALUE(Prompt_Quantity,FALSE),)	
126		= SET.VALUE(Reorder_Quantity,K125)	
127		= END.IF()	
128		=END.IF()	
129			
130	Order	=Reorder_Quantity*Reorder+Amount_Order*Make_Order	Determine Order Quantity
131			
132	Amount_Received	=RUN('WARE-1.XLM'!Warehouse,FALSE)	Place Order at Warehouse
133			
134		=IF(Ran_Lead_Time=1,Amount_Received,FALSE)	Account for Lead Time
135		=IF(Ran_Lead_Time=2,Amount_Received,FALSE)	
136		=IF(Ran_Lead_Time=3,Amoun: Received,FALSE)	
137		=IF(Ran_Lead_Time=4,Amount_Received,FALSE)	
138		=IF(Ran_Lead_Time=5,Amount_Received,FALSE)	
139			
140	Shipment	=B141+B134	Count Down Orders by Day
141		=B142+B135	
142		=B143+B136	
143		=B144+B137	
144		=B138	



	A	B	C
1	*names*	**Commands**	*comments*
2			
145			
146		=IF(MASTER.XLM!P_Start_Iteration=TRUE,GOTO(B153),)	Skip Output if Startup
147			
148	Total_Shortage_Cost	=Back_Order*Shortage_Cost+B148	Determine Costs
149	Total_Holding_Cost	=Current_Inventory*Holding_Cost+B149	
150	Total_Order_Cost	=(Make_Order+Reorder)*Cost_Order+B150	
151	Total_Cost	=B148+B149+B150	
152			
153		=RUN(DEALER.XLM!Dealer_Output)	Record Desired Variables
154			
155		=RETURN()	
156			
157			**Output Routine**
158			
159	Dealer_Output	=FORMULA(Day,OFFSET($T$4,Day,0))	**Day**
160		=FORMULA(Current_Inventory,OFFSET($T$4,Day,1))	**Inventory**
161		=FORMULA(Back_Order,OFFSET($T$4,Day,2))	**Back_Order**
162		=FORMULA(Total_Shortage_Cost,OFFSET($T$4,Day,3))	**Total_Shortage_Cost**
163		=FORMULA(Total_Holding_Cost,OFFSET($T$4,Day,4))	**Total_Holding_Cost**
164		=FORMULA(Total_Order_Cost,OFFSET($T$4,Day,5))	**Total_Order_Cost**
165		=FORMULA(Total_Cost,OFFSET($T$4,Day,6))	**Total_Cost**
166		=FORMULA(DEMAND,OFFSET($T$4,Day,7))	**Demand**
167		=FORMULA(Ran_Lead_Time,OFFSET($T$4,Day,8))	**Random Lead Time**
168		=FORMULA(Order,OFFSET($T$4,Day,9))	**Amount_Order**
169		=FORMULA(Make_Order,OFFSET($T$4,Day,10))	**Make_Order**
170		=FORMULA(Reorder,OFFSET($T$4,Day,11))	**Reorder**
171		=FORMULA(Shipment,OFFSET($T$4,Day,12))	**Shipment**
172			
173		=RETURN()	

	E	F	G	H	I	J	K	L
1	DIALOGS							
2	type	x	y	wide	high	text	init/result	*names*
50	Input							
51		242	74	536	336	Initial Dealer Data		
52	1	435	20	88		DONE		
53	14	21	15	199	142	Daily Demand		
54	11						5	Daily_Demand
55	12		46		21	0		
56	12					1		
57	12					2		
58	12					3		
59	12					4		
60	5	44	32			Demand		
61	8	75	48	46			4	Demand_0
62	8	75	69	46			5	Demand_1
63	8	75	90	46			6	Demand_2
64	8	75	110	46			7	Demand_3
65	8	75	130	46			8	Demand_4
66	5	134	32			Frequency		
67	8	149	48	46			20	Demand_Freq_0
68	8	149	69	46			20	Demand_Freq_1
69	8	149	90	46			20	Demand_Freq_2
70	8	149	110	46			20	Demand_Freq_3
71	8	149	130	46			20	Demand_Freq_4
72	14	16	178	205	140	Lead Time		
73	11						5	Lead_Time
74	12		202		21	1		
75	12					2		
76	12					3		
77	12					4		
78	12					5		
79	5	93	193			Frequency		
80	8	82	206	100			20	Lead_Time_Freq_1
81	8	82	226	100			20	Lead_Time_Freq_2
82	8	82	245	100			20	Lead_Time_Freq_3
83	8	82	265	100			20	Lead_Time_Freq_4
84	8	82	285	100			20	Lead_Time_Freq_5
85	5	251	97			Beginning Inventory		
86	7	456	97	60			50	Beginning_Inventory
87	14	241	120	277	104	Ordering Information		
88	13					Prompt for Reorder	FALSE	Prompt_Reorder
89	13					Prompt for Reorder Quantity	FALSE	Prompt_Quantity
90	5	272	172			Reorder Point		
91	7	424	170	73			25	Reorder_Point
92	5	272	192			Reorder Quantity		
93	7	424	193	74			25	Reorder_Quantity
94	14	243	231	275	88	Costs		
95	5	252	248			Holding Cost/Unit		
96	8	409	246	100			0.1	Holding_Cost
97	5	253	270			Cost/Order		
98	8	409	267	100			20	Cost_Order
99	5	253	291			Shortage Cost/Unit		
100	8	409	288	100			50	Shortage_Cost
101	2	433	53	88		Quit		

	E	F	G	H	I	J	K	L
1	DIALOGS							
2	type	x	y	wide	high	text	init/result	*names*
107	Reorder							
108		250	162	516	152	Dealer Order		
109	13	90	84		35	Do You Wish to Make an Order?	FALSE	Make_Order
110	5	90	119			Amount of Order?		
111	7	310	118	60			25	Amount_Order
112	3	408	9	88		DONE		
113	2	408	40	88		Quit		
114	17	22	42			1		
115	5	90	14			It is day		
116	7	310	13	60			17	
117	5	90	39			Today's Demand		
118	7	310	38	60			7	
119	5	90	66			Current Inventory Level		
120	7	310	65	60			48	
121								
122	Order							
123		250	162	516	152	Dealer Order		
124	5	90	119			Amount of Order?		
125	7	310	118	60			25	
126	3	408	40	88		DONE		
127	2	408	70	88		Quit		
128	5	90	14			Inventory is at or Below the Reorder P		
129	5	90	39			It is day		
130	7	310	38	60			1	
131	5	90	66			Today's Demand		
132	7	310	65	60			2	
133	5	90	93			Current Inventory		
134	7	310	92	60			7	
135	17	22	42			1		

	N	O	P	Q	R
1	TABLES				
2					
50	Demand Table	Daily	Frequency	Relative Freq	Cumulative Freq
51	=GOTO(O51)	=J55	=Demand_Freq_0	=P51/SUM(P$51:P$55)	=Q51
52		=J56	=Demand_Freq_2	=P52/SUM(P$51:P$55)	=Q52+R51
53		=J57	=Demand_Freq_2	=P53/SUM(P$51:P$55)	=Q53+R52
54		=J58	=Demand_Freq_3	=P54/SUM(P$51:P$55)	=Q54+R53
55		=J59	=Demand_Freq_4	=P55/SUM(P$51:P$55)	=Q55+R54
56		=GOTO(P51	=GOTO(Q51)	=GOTO(R51)	=GOTO(B64)
57					
58	Lead Time Table	Daily	Frequency	Relative Freq	Cumulative Freq
59	=GOTO(O59)	=J74	=Lead_Time_Freq_1	=P59/SUM(P$59:P$63)	=Q59
60		=J75	=Lead_Time_Freq_2	=P60/SUM(P$59:P$63)	=Q60+R59
61		=J76	=Lead_Time_Freq_3	=P61/SUM(P$59:P$63)	=Q61+R60
62		=J77	=Lead_Time_Freq_4	=P62/SUM(P$59:P$63)	=Q62+R61
63		=J78	=Lead_Time_Freq_5	=P63/SUM(P$59:P$63)	=Q63+R62
64		=GOTO(P59	=GOTO(Q59)	=GOTO(R59)	=GOTO(B66)

WAREHOUSE #1 MANAGEMENT MACRO

	A	B	C
1	*names*	**Commands**	*comments*
2			
3		**Summary Information**	
4	*Title:*	Warehouse # 1 Management Macro	
5	*Version:*	v1.0	
6	*Author*	Dixon Hicks	
7	*Corporation:*	Naval Postgraduate School	
8	*Creation Date:*	Wednesday, January 20, 1993	
9			
10	*CommandWindow*	**CommandWindow**	
11	Warehouse	=IF(MASTER.XLM!First_Iteration=TRUE)	Start Warehouse Macro
12			
13		= SET.VALUE(Beginning_Inventory,MASTER.XLM!Warehouse_BI)	Set Initial Variable
14		= SET.VALUE(Prompt_Reorder,FALSE)	
15		= SET.VALUE(Prompt_Quantity,FALSE)	
16		= SET.VALUE(Reorder_Point,MASTER.XLM!Warehouse_RL)	
17		= SET.VALUE(Reorder_Quantity,MASTER.XLM!Warehouse_RQ)	
18		= SET.VALUE(Holding_Cost,MASTER.XLM!Warehouse_HC)	
19		= SET.VALUE(Cost_Order,MASTER.XLM!Warehouse_OC)	
20		= SET.VALUE(Shortage_Cost,MASTER.XLM!Warehouse_SC)	
21		= SET.VALUE(Lead_Time,5)	
22		= SET.VALUE(Day,0)	
23		= SET.VALUE(Shipment,0)	
24		= SET.VALUF(Order,0)	
25		= SET.VALUE(Back_Order,0)	
26		= SET.VALUE(Total_Cost,0)	
27		= SET.VALUE(Total_Holding_Cost,0)	
28		= SET.VALUE(Total_Order_Cost,0)	
29		= SET.VALUE(Total_Shortage_Cost,0)	
30		= SET.VALUE(Received_Order,0)	
31		= SET.VALUE(Reorder,FALSE)	
32		= SET.VALUE(Make_Order,FALSE)	
33		= SET.VALUE(B128,0)	
34		= SET.VALUE(B129,0)	
35		= SET.VALUE(B130,0)	
36		= SET.VALUE(B131,0)	
37		= SET.VALUE(B132,0)	
38			
39		= IF(MASTER.XLM!Prompt_Individual=TRUE)	
40		= DIALOG.BOX(Input_Box)	User Choose Variables
41		= IF(B40=FALSE)	End Simulation by user
42	QUIT	= ALERT("Simulation Cancelled",3)	
43		= MESSAGE(FALSE)	
44		= ACTIVATE("MASTER.XLS")	
45		= HALT()	
46		= END.IF()	
47		= END.IF()	
48			
49		= SET.VALUE(Current_Inventory,Beginning_Inventory)	Set Current Inventory
50		= GOTO(Lead_Time_Table)	Determine Relative Frequency
51			
52		= SET.NAME("Output_Data",$T$4:OFFSET($T$4,(MASTER.XLM!Total_Iterations+1),12))	Clear Output Data
53		= FORMULA.GOTO(Output_Data,T4)	
54		= CLEAR(3)	
55		= VSCROLL(1,TRUE)	
56		= RUN("WARE-1.XLM"!Warehouse_Output)	Record Initial Variables
57		=END.IF()	
58			
59			
60			Begin Routine
61			
62	Day	=MASTER.XLM!Day	Counter for Reference
63			
64		=RAND()	Determine Lead Time
66		=IF(B$64>(VLOOKUP(1,Lead_Time_Table,4)),2,1)	
66		=IF(B$64>(VLOOKUP(2,Lead_Time_Table,4)),3,1)	
67		=IF(B$64>(VLOOKUP(3,Lead_Time_Table,4)),4,1)	
68		=IF(B$64>(VLOOKUP(4,Lead_Time_Table,4)),5,1)	
69	Ran_Lead_Time	=MAX(B65:B68)	

	A	B	C
1	*names*	**Commands**	comments
2			
70			
71		=Current_Inventory-Back_Order	Fill Backorder if Possible
72		=IF(B71>=0,0,ABS(B71))	Determine Size of Backorder
73		=IF(B71>0,B71,0)	Ensure Inventory not Negative
74		=Current_Inventory-B73	Determine Change in Inventory
75			
76		=B73-DEALER.XLM!Order	Fill Demand if Possible
77		=IF(B76>=0,B76,0)	Ensure Inventory not Negative
78		=B73-B77	Determine Change in Inventory
79	Shipment	=B74+B78	Determine Amount Shipped
80	Back_Order	=IF(B76<0,B72+ABS(B76),B72)	Adjust Backorder Level
81			
82	Current_Inventory	=B77+Received_Order	Receive Shipment if Present
83			
84		=IF(MASTER.XLM!P_Start_Iteration=TRUE,GOTO(Reorder),)	Skip Reorder ? if Startup
85			
86		=IF(Prompt_Reorder=TRUE)	Reorder & Quantity?
87		= SET.VALUE(Reorder,FALSE)	
88		= SET.VALUE(K98,Current_Inventory-Back_Order)	
89		= SET.VALUE(K94,Day)	
90		= SET.VALUE(K96,DEALER.XLM!Order)	
91		= SET.VALUE(Make_Order,FALSE)	
92		= DIALOG.BOX(Reorder_Box)	
93		= IF(B92=FALSE,SET.VALUE(Prompt_Reorder,FALSE),)	
94		= GOTO(Order)	
95		=END.IF()	
96			
97	Reorder	=IF(Current_Inventory<Reorder_Point,TRUE,FALSE)	
98			
99		=IF(MASTER.XLM!P_Start_Iteration=TRUE,GOTO(Order),)	Skip Quantity ? if Startup
100			
101		=IF(Reorder=TRUE)	
102		= IF(Prompt_Quantity=TRUE)	Quantity?
103		= SET.VALUE(K107,Reorder_Quantity)	
104		= SET.VALUE(K111,Day)	
105		= SET.VALUE(K113,DEALER.XLM!Order)	
106		= SET.VALUE(K115,Current_Inventory-Back_Order)	
107		= DIALOG.BOX(Order_Box)	
108		= IF(B107=FALSE,SET.VALUE(Prompt_Quantity,FALSE),)	
109		= SET.VALUE(Reorder_Quantity,K107)	
110		= END.IF()	
111		=END.IF()	
112			
113	Order	=Reorder_Quantity*Reorder+Amount_Order*Make_Order	Determine Order Quantity
114			
115		=IF(MASTER.XLM!Warehouse_Number=1)	Execute appropriate Warehouse
116	Amount_Received	= RUN(FACTORY.XLM!Factory,FALSE)	or Factory Macro to place
117		=ELSE.IF(MASTER.XLM!Warehouse_Number>=2)	order
118		= RUN('WARE-2.XLM'!Warehouse,FALSE)	
119		= SET.VALUE(Amount_Received,B118)	
120		=END.IF()	
121			
122		=IF(Ran_Lead_Time=1,Amount_Received,FALSE)	Account for Lead Time
123		=IF(Ran_Lead_Time=2,Amount_Received,FALSE)	
124		=IF(Ran_Lead_Time=3,Amount_Received,FALSE)	
125		=IF(Ran_Lead_Time=4,Amount_Received,FALSE)	
126		=IF(Ran_Lead_Time=5,Amount_Received,FALSE)	
127			
128	Received_Order	=B129+B122	Count Down Orders by Day
129		=B130+B123	
130		=B131+B124	
131		=B132+B125	
132		=B126	
133			
134		=IF(MASTER.XLM!P_Start_Iteration=TRUE,GOTO(B141),)	Skip & Output if Startup
135			
136	Total_Shortage_Cost	=Back_Order*Shortage_Cost+B136	Determine Costs
137	Total_Holding_Cost	=Current_Inventory*Holding_Cost+B137	
138	Total_Order_Cost	=(Make_Order+Reorder)*Cost_Order+B138	
139	Total_Cost	=B136+B137+B138	
140			

	A	B	C
1	*names*	**Commands**	*comments*
2			
141		=RUN('WARE-1.XLM'!Warehouse_Output)	Record Desired Variables
142			
143		=RETURN(Shipment)	
144			
145			**Output Routine**
146			
147	Warehouse_Output	=FORMULA(Day,OFFSET($T$4,Day,0))	**Day**
148		=FORMULA(Current_Inventory,OFFSET($T$4,Day,1))	**Inventory**
149		=FORMULA(Back_Order,OFFSET($T$4,Day,2))	**Back_Order**
150		=FORMULA(Total_Shortage_Cost,OFFSET($T$4,Day,3))	**Total_Shortage_Cost**
151		=FORMULA(Total_Holding_Cost,OFFSET($T$4,Day,4))	**Total_Holding_Cost**
152		=FORMULA(Total_Order_Cost,OFFSET($T$4,Day,5))	**Total_Order_Cost**
153		=FORMULA(Total_Cost,OFFSET($T$4,Day,6))	**Total_Cost**
154		=FORMULA(DEALER.XLM!Order,OFFSET($T$4,Day,7))	**Demand**
155		=FORMULA(Ran_Lead_Time,OFFSET($T$4,Day,8))	**Random Lead Time**
156		=FORMULA(Amount_Order,OFFSET($T$4,Day,9))	**Amount_Order**
157		=FORMULA(Make_Order,OFFSET($T$4,Day,10))	**Make_Order**
158		=FORMULA(Reorder,OFFSET($T$4,Day,11))	**Reorder**
159		=FORMULA(Shipment,OFFSET($T$4,Day,12))	**Shipment**
160			
161		=RETURN()	

	E	F	G	H	I	J	K	L
1	DIALOGS							
2	type	x	y	wide	high	text	init/result	*names*
38	Input							
39		243	102	536	265	Warehouse #1 Initial Data		
40	1	423	11	88		DONE		
41	5	21	10			Beginning Inventory		
42	7	226	10	60			50	Beginning_Inventory
43	14	11	34	277	104	Ordering Information		
44	13					Prompt for Reorder	FALSE	Prompt_Reorder
45	13					Prompt for Reorder Quantity	FALSE	Prompt_Quantity
46	5	42	86			Reorder Point		
47	7	194	84	73			25	Reorder_Point
48	5	42	106			Reorder Quantity		
49	7	194	107	74			25	Reorder_Quantity
50	14	12	144	279	104	Costs		
51	5	21	161			Holding Cost/Unit		
52	8	178	159	100			0.1	Holding_Cost
53	5	22	183			Cost/Order		
54	8	178	180	100			20	Cost_Order
55	5	22	204			Shortage Cost/Unit		
56	8	178	201	100			50	Shortage_Cost
57	14	309	109	205	140	Lead Time		
58	11						5	Lead_Time
59	12		133		20	1		
60	12					2		
61	12					3		
62	12					4		
63	12					5		
64	5	386	124			Frequency		
65	8	375	137	100			20	Lead_Time_Freq_1
66	8	375	157	100			20	Lead_Time_Freq_2
67	8	375	176	100			20	Lead_Time_Freq_3
68	8	375	196	100			20	Lead_Time_Freq_4
69	8	375	216	100			20	Lead_Time_Freq_5
70	2	423	42	88		Quit		

	E	F	G	H	I	J	K	L
1	DIALOGS							
2	type	x	y	wide	high	text	init/result	*names*
87	Reorder							
88		250	175	516	125	Warehouse #1 Order		
89	13	114	67		35	Do You Wish to Make an Order?	FALSE	Make_Order
90	5	114	100			Amount of Order?		
91	7	318	99	60			50	Amount_Order
92	3	408	13	88		DONE		
93	5	114	9			The Day is		
94	7	318	8	60			17	
95	5	114	32			Today's Demand		
96	7	318	31	60			0	
97	5	114	55			Current Inventory Level		
98	7	318	54	60			50	
99	17	28	29			1		
100	2	408	40	88		Quit		
101								
102								
103								
104	Order							
105		250	175	516	125	Warehouse #1 Order		
106	5	114	100			Amount of Order?		
107	7	318	99	60			25	
108	3	408	13	88		DONE		
109	5	114	79			Inventory is at or Below the Reorder P		
110	5	114	9			The Day is		
111	7	318	8	60			1	
112	5	114	32			Today's Demand		
113	7	318	31	60			2	
114	5	114	55			Current Inventory		
115	7	318	54	60			2	
116	2	408	40	88		Quit		
117	17	28	29			1		

WAREHOUSE MANAGEMENT MACRO
WAREHOUSE #2 MANAGEMENT MACRO

	A	B	C
1	*names*	**Commands**	*comments*
2			
3		**Summary Information**	
4	*Title:*	Warehouse # 2 Management Macro	
5	*Version:*	v1.0	
6	*Author*	Dixon Hicks	
7	*Corporation:*	Naval Postgraduate School	
8	*Creation Date:*	Wednesday, January 20, 1993	
9			
10	*CommandWindow*	**CommandWindow**	
11	Warehouse	=IF(MASTER.XLM!First_Iteration=TRUE)	Start Warehouse Macro
12			
13		= SET.VALUE(Beginning_Inventory,MASTER.XLM!Warehouse_BI)	Set Initial Variable
14		= SET.VALUE(Prompt_Reorder,FALSE)	
15		= SET.VALUE(Prompt_Quantity,FALSE)	
16		= SET.VALUE(Reorder_Point,MASTER.XLM!Warehouse_RL)	
17		= SET.VALUE(Reorder_Quantity,MASTER.XLM!Warehouse_RQ)	
18		= SET.VALUE(Holding_Cost,MASTER.XLM!Warehouse_HC)	
19		= SET.VALUE(Cost_Order,MASTER.XLM!Warehouse_OC)	
20		= SET.VALUE(Shortage_Cost,MASTER.XLM!Warehouse_SC)	
21		= SET.VALUE(Lead_Time,5)	
22		= SET.VALUE(Day,0)	
23		= SET.VALUE(Shipment,0)	
24		= SET.VALUE(Order,0)	
25		= SET.VALUE(Back_Order,0)	
26		= SET.VALUE(Total_Cost,0)	
27		= SET.VALUE(Total_Holding_Cost,0)	
28		= SET.VALUE(Total_Order_Cost,0)	
29		= SET.VALUE(Total_Shortage_Cost,0)	
30		= SET.VALUE(Received_Order,0)	
31		= SET.VALUE(Reorder,FALSE)	
32		= SET.VALUE(Make_Order,FALSE)	
33		= SET.VALUE(B128,0)	
34		= SET.VALUE(B129,0)	
35		= SET.VALUE(B130,0)	
36		= SET.VALUE(B131,0)	
37		= SET.VALUE(B132,0)	
38			
39		= IF(MASTER.XLM!Prompt_Individual=TRUE)	
40		= DIALOG.BOX(Input_Box)	User Choose Variables
41		= IF(B40=FALSE)	End Simulation by user
42	QUIT	= ALERT("Simulation Cancelled",3)	
43		= MESSAGE(FALSE)	
44		= ACTIVATE("MASTER.XLS")	
45		= HALT()	
46		= END.IF()	
47		= END.IF()	
48			
49		= SET.VALUE(Current_Inventory,Beginning_Inventory)	Set Current Inventory
50		= GOTO(Lead_Time_Table)	Determine Relative Frequency
51			
52		= SET.NAME("Output_Data",$T$4:OFFSET($T$4,(MASTER.XLM!Total_Iterations+1),12))	Clear Output Data
53		= FORMULA.GOTO(Output_Data,T4)	
54		= CLEAR(3)	
55		= VSCROLL(1,TRUE)	
56		= RUN("WARE-2.XLM"!Warehouse_Output)	Record Initial Variables
57		=END.IF()	
58			
59			
60			**Begin Routine**
61			
62	Day	=MASTER.XLM!Day	Counter for Reference
63			
64		=RAND()	Determine Lead Time
65		=IF(B$64>(VLOOKUP(1,Lead_Time_Table,4)),2,1)	
66		=IF(B$64>(VLOOKUP(2,Lead_Time_Table,4)),3,1)	
67		=IF(B$64>(VLOOKUP(3,Lead_Time_Table,4)),4,1)	
68		=IF(B$64>VLOOKUP(4,Lead_Time_Table,4)),5,1)	
69	Ran_Lead_Time	=MAX(B65:B68)	

	A	B	C
1	*names*	**Commands**	*comments*
2			
70			
71		=Current_Inventory-Back_Order	Fill Backorder if Possible
72		=IF(B71>=0,0,ABS(B71))	Determine Size of Backorder
73		=IF(B71>0,B71,0)	Ensure Inventory not Negative
74		=Current_Inventory-B73	Determine Change in Inventory
75			
76		=B73-'WARE-1.XLM'!Order	Fill Demand if Possible
77		=IF(B76>=0,B76,0)	Ensure Inventory not Negative
78		=B73-B77	Determine Change in Inventory
79	Shipment	=B74+B78	Determine Amount Shipped
80	Back_Order	=IF(B76<0,B72+ABS(B76),B72)	Adjust Backorder Level
81			
82	Current_Inventory	=B77+Received_Order	Receive Shipment if Present
83			
84		=IF(MASTER.XLM!P_Start_Iteration=TRUE,GOTO(Reorder),)	Skip Reorder ? if Startup
85			
86		=IF(Prompt_Reorder=TRUE)	Reorder & Quantity?
87		= SET.VALUE(Reorder,FALSE)	
88		= SET.VALUE(K98,Current_Inventory-Back_Order)	
89		= SET.VALUE(K94,Day)	
90		= SET.VALUE(K96,'WARE-1.XLM'!Order)	
91		= SET.VALUE(Make_Order,FALSE)	
92		= DIALOG.BOX(Reorder_Box)	
93		= IF(B92=FALSE,SET.VALUE(Prompt_Reorder,FALSE),)	
94		= GOTO(Order)	
95		=END.IF()	
96			
97	Reorder	=IF(Current_Inventory<Reorder_Point,TRUE,FALSE)	
98			
99		=IF(MASTER.XLM!P_Start_Iteration=TRUE,GOTO(Order),)	Skip Quantity ? if Startup
100			
101		=IF(Reorder=TRUE)	
102		= IF(Prompt_Quantity=TRUE)	Quantity?
103		= SET.VALUE(K107,Reorder_Quantity)	
104		= SET.VALUE(K111,Day)	
105		= SET.VALUE(K113,'WARE-1.XLM'!Order)	
106		= SET.VALUE(K115,Current_Inventory-Back_Order)	
107		= DIALOG.BOX(Order_Box)	
108		= IF(B107=FALSE,SET.VALUE(Prompt_Quantity,FALSE),)	
109		= SET.VALUE(Reorder_Quantity,K107)	
110		= END.IF()	
111		=END.IF()	
112			
113	Order	=Reorder_Quantity*Reorder+Amount_Order*Make_Order	Determine Order Quantity
114			
115		=IF(MASTER.XLM!Warehouse_Number=2)	Execute appropriate Warehouse
116	Amount_Received	= RUN(FACTORY.XLM!Factory,FALSE)	or Factory Macro to place
117		=ELSE.IF(MASTER.XLM!Warehouse_Number>=3)	order
118		= RUN('WARE-3.XLM'!Warehouse,FALSE)	
119		= SET.VALUE(Amount_Received,B118)	
120		=END.IF()	
121			
122		=IF(Ran_Lead_Time=1,Amount_Received,FALSE)	Account for Lead Time
123		=IF(Ran_Lead_Time=2,Amount_Received,FALSE)	
124		=IF(Ran_Lead_Time=3,Amount_Received,FALSE)	
125		=IF(Ran_Lead_Time=4,Amount_Received,FALSE)	
126		=IF(Ran_Lead_Time=5,Amount_Received,FALSE)	
127			
128	Received_Order	=B129+B122	Count Down Orders by Day
129		=B130+B123	
130		=B131+B124	
131		=B132+B125	
132		=B126	
133			
134		=IF(MASTER.XLM!P_Start_Iteration=TRUE,GOTO(B141),)	Skip & Output if Startup
135			
136	Total_Shortage_Cost	=Back_Order*Shortage_Cost+B136	Determine Costs
137	Total_Holding_Cost	=Current_Inventory*Holding_Cost+B137	
138	Total_Order_Cost	=(Make_Order+Reorder)*Cost_Order+B138	
139	Total_Cost	=B136+B137+B138	
140			

	A	B	C
1	*names*	**Commands**	*comments*
2			
141		=RUN('WARE-2.XLM'!Warehouse_Output)	Record Desired Variables
142			
143		=RETURN(Shipment)	
144			
145			**Output Routine**
146			
147	Warehouse_Output	=FORMULA(Day,OFFSET($T$4,Day,0))	**Day**
148		=FORMULA(Current_Inventory,OFFSET($T$4,Day,1))	**Inventory**
149		=FORMULA(Back_Order,OFFSET($T$4,Day,2))	**Back_Order**
150		=FORMULA(Total_Shortage_Cost,OFFSET($T$4,Day,3))	**Total_Shortage_Cost**
151		=FORMULA(Total_Holding_Cost,OFFSET($T$4,Day,4))	**Total_Holding_Cost**
152		=FORMULA(Total_Order_Cost,OFFSET($T$4,Day,5))	**Total_Order_Cost**
153		=FORMULA(Total_Cost,OFFSET($T$4,Day,6))	**Total_Cost**
154		=FORMULA('WARE-1.XLM'!Order,OFFSET($T$4,Day,7))	**Demand**
155		=FORMULA(Ran_Lead_Time,OFFSET($T$4,Day,8))	**Random Lead Time**
156		=FORMULA(Amount_Order,OFFSET($T$4,Day,9))	**Amount_Order**
157		=FORMULA(Make_Order,OFFSET($T$4,Day,10))	**Make_Order**
158		=FORMULA(Reorder,OFFSET($T$4,Day,11))	**Reorder**
159		=FORMULA(Shipment,OFFSET($T$4,Day,12))	**Shipment**
160			
161		=RETURN()	

WAREHOUSE #3 MANAGEMENT MACRO

	A	B	C
1	*names*	**Commands**	*comments*
2			
3		**Summary Information**	
4	*Title:*	Warehouse # 3 Management Macro	
5	*Version:*	v1.0	
6	*Author*	Dixon Hicks	
7	*Corporation:*	Naval Postgraduate School	
8	*Creation Date:*	Wednesday, January 20, 1993	
9			
10	*CommandWindow*	**CommandWindow**	
11	**Warehouse**	=IF(MASTER.XLM!First_Iteration=TRUE)	Start Warehouse Macro
12			
13		= SET.VALUE(Beginning_Inventory,MASTER.XLM!Warehouse_BI)	Set Initial Variable
14		= SET.VALUE(Prompt_Reorder,FALSE)	
15		= SET.VALUE(Prompt_Quantity,FALSE)	
16		= SET.VALUE(Reorder_Point,MASTER.XLM!Warehouse_RL)	
17		= SET.VALUE(Reorder_Quantity,MASTER.XLM!Warehouse_RQ)	
18		= SET.VALUE(Holding_Cost,MASTER.XLM!Warehouse_HC)	
19		= SET VALUE(Cost_Order,MASTER.XLM!Warehouse_OC)	
20		= SET.VALUE(Shortage_Cost,MASTER.XLM!Warehouse_SC)	
21		= SET.VALUE(Lead_Time,5)	
22		= SET.VALUE(Day,0)	
23		= SET.VALUE(Shipment,0)	
24		= SET.VALUE(Order,0)	
25		= SET.VALUE(Back_Order,0)	
26		= SET.VALUE(Total_Cost,0)	
27		= SET.VALUE(Total_Holding_Cost,0)	
28		= SET.VALUE(Total_Order_Cost,0)	
29		= SET.VALUE(Total_Shortage_Cost,0)	
30		= SET.VALUE(Received_Order,0)	
31		= SET.VALUE(Reorder,FALSE)	
32		= SET.VALUE(Make_Order,FALSE)	
33		= SET.VALUE(B128,0)	
34		= SET.VALUE(B129,0)	
35		= SET.VALUE(B130,0)	
36		= SET.VALUE(B131,0)	
37		= SET.VALUE(B132,0)	
38			
39		= IF(MASTER.XLM!Prompt_Individual=TRUE)	
40		= DIALOG.BOX(Input_Box)	User Choose Variables
41		= IF(B40=FALSE)	End Simulation by user
42	QUIT	= ALERT("Simulation Cancelled",3)	
43		= MESSAGE(FALSE)	
44		= ACTIVATE("MASTER.XLS")	
45		= HALT()	
46		= END.IF()	
47		= END.IF()	
48			
49		= *SET.VALUE(Current_Inventory,Beginning_Inventory)*	Set Current Inventory
50		= GOTO(Lead_Time_Table)	Determine Relative Frequency
51			
52		= SET.NAME("Output_Data",$T$4:OFFSET($T$4,(MASTER.XLM!Total_Iterations+1),12))	Clear Output Data
53		= *FORMULA.GOTO(Output_Data,T4)*	
54		= CLEAR(3)	
55		= *VSCROLL(1,TRUE)*	
56		= RUN('WARE-3.XLM'!Warehouse_Output)	Record Initial Variables
57		*=END.IF()*	
58			
59			
60			**Begin Routine**
61			
62	Day	=MASTER.XLM!Day	Counter for Reference
63			
64		=RAND()	Determine Lead Time
65		=IF(B$64>(VLOOKUP(1,Lead_Time_Table,4)),2,1)	
66		=IF(B$64>(VLOOKUP(2,Lead_Time_Table,4)),3,1)	
67		=IF(B$64>(VLOOKUP(3,Lead_Time_Table,4)),4,1)	
68		=IF(B$64>(VLOOKUP(4,Lead_Time_Table,4)),5,1)	
69	*Ran_Lead_Time*	=MAX(B65:B68)	

	A	B	C
1	*names*	**Commands**	*comments*
2			
70			
71		=Current_Inventory-Back_Order	Fill Backorder if Possible
72		=IF(B71>=0,0,ABS(B71))	Determine Size of Backorder
73		=IF(B71>0,B71,0)	Ensure Inventory not Negative
74		=Current_Inventory-B73	Determine Change in Inventory
75			
76		=B73-'WARE-2.XLM'!Order	Fill Demand if Possible
77		=IF(B76>=0,B76,0)	Ensure Inventory not Negative
78		=B73-B77	Determine Change in Inventory
79	Shipment	=B74+B78	Determine Amount Shipped
80	Back_Order	=IF(B76<0,B72+ABS(B76),B72)	Adjust Backorder Level
81			
82	Current_Inventory	=B77+Received_Order	Receive Shipment if Present
83			
84		=IF(MASTER.XLM!P_Start_Iteration=TRUE,GOTO(Reorder),)	Skip Reorder ? if Startup
85			
86		=IF(Prompt_Reorder=TRUE)	Reorder & Quantity?
87		= SET.VALUE(Reorder,FALSE)	
88		= SET.VALUE(K98,Current_Inventory-Back_Order)	
89		= SET.VALUE(K94,Day)	
90		= SET.VALUE(K96,'WARE-2.XLM'!Order)	
91		= SET.VALUE(Make_Order,FALSE)	
92		= DIALOG.BOX(Reorder_Box)	
93		= IF(B92=FALSE,SET.VALUE(Prompt_Reorder,FALSE),)	
94		= GOTO(Order)	
95		=END.IF()	
96			
97	Reorder	=IF(Current_Inventory<Reorder_Point,TRUE,FALSE)	
98			
99		=IF(MASTER.XLM!P_Start_Iteration=TRUE,GOTO(Order),)	Skip Quantity ? if Startup
100			
101		=IF(Reorder=TRUE)	
102		= IF(Prompt_Quantity=TRUE)	Quantity?
103		= SET.VALUE(K107,Reorder_Quantity)	
104		= SET.VALUE(K111,Day)	
105		= SET.VALUE(K113,'WARE-2.XLM'!Order)	
106		= SET.VALUE(K115,Current_Inventory-Back_Order)	
107		= DIALOG.BOX(Order_Box)	
108		= IF(B107=FALSE,SET.VALUE(Prompt_Quantity,FALSE),)	
109		= SET.VALUE(Reorder_Quantity,K107)	
110		= END.IF()	
111		=END.IF()	
112			
113	Order	=Reorder_Quantity*Reorder+Amount_Order*Make_Order	Determine Order Quantity
114			
115			Execute appropriate Warehouse
116	Amount_Received	= RUN(FACTORY.XLM!Factory,FALSE)	or Factory Macro to place
117			order
118			
119			
120			
121			
122		=IF(Ran_Lead_Time=1,Amount_Received,FALSE)	Account for Lead Time
123		=IF(Ran_Lead_Time=2,Amount_Received,FALSE)	
124		=IF(Ran_Lead_Time=3,Amount_Received,FALSE)	
125		=IF(Ran_Lead_Time=4,Amount_Received,FALSE)	
126		=IF(Ran_Lead_Time=5,Amount_Received,FALSE)	
127			
128	Received_Order	=B129+B122	Count Down Orders by Day
129		=B130+B123	
130		=B131+B124	
131		=B132+B125	
132		=B126	
133			
134		=IF(MASTER.XLM!P_Start_Iteration=TRUE,GOTO(B141),)	Skip & Output if Startup
135			
136	Total_Shortage_Cost	=Back_Order*Shortage_Cost+B136	Determine Costs
137	Total_Holding_Cost	=Current_Inventory*Holding_Cost+B137	
138	Total_Order_Cost	=(Make_Order+Reorder)*Cost_Order+B138	
139	Total_Cost	=B136+B137+B138	
140			

	A	B	C
1	*names*	**Commands**	*comments*
2			
141		=RUN('WARE-3.XLM'!Warehouse_Output)	Record Desired Variables
142			
143		=RETURN(Shipment)	
144			
145			**Output Routine**
146			
147	Warehouse_Output	=FORMULA(Day,OFFSET($T$4,Day,0))	**Day**
148		=FORMULA(Current_Inventory,OFFSET($T$4,Day,1))	**Inventory**
149		=FORMULA(Back_Order,OFFSET($T$4,Day,2))	**Back_Order**
150		=FORMULA(Total_Shortage_Cost,OFFSET($T$4,Day,3))	**Total_Shortage_Cost**
151		=FORMULA(Total_Holding_Cost,OFFSET($T$4,Day,4))	**Total_Holding_Cost**
152		=FORMULA(Total_Order_Cost,OFFSET($T$4,Day,5))	**Total_Order_Cost**
153		=FORMULA(Total_Cost,OFFSET($T$4,Day,6))	**Total_Cost**
154		=FORMULA('WARE-2.XLM'!Order,OFFSET($T$4,Day,7))	**Demand**
155		=FORMULA(Ran_Lead_Time,OFFSET($T$4,Day,8))	**Random Lead Time**
156		=FORMULA(Amount_Order,OFFSET($T$4,Day,9))	**Amount_Order**
157		=FORMULA(Make_Order,OFFSET($T$4,Day,10))	**Make_Order**
158		=FORMULA(Reorder,OFFSET($T$4,Day,11))	**Reorder**
159		=FORMULA(Shipment,OFFSET($T$4,Day,12))	**Shipment**
160			
161		=RETURN()	

FACTORY MANAGEMENT MACRO

	A	B	C
1	*names*	**Commands**	*comments*
2			
3		**Summary Information**	
4	*Title:*	Factory Management Macro	
5	*Version:*	v1.0	
6	*Author*	Dixon Hicks	
7	*Corporation:*	Naval Postgraduate School	
8	*Creation Date:*	Wednesday, January 20, 1993	
9			
10	*CommandWindow*	**CommandWindow**	
11	Factory	=IF(MASTER.XLM!First_Iteration=TRUE)	Start Factory Routine
12			
13		= SET.VALUE(Beginning_Inventory,MASTER.XLM!Factory_BI)	Set Initial Variable
14		= SET.VALUE(Prompt_Begin_Production,FALSE)	
15		= SET.VALUE(Prompt_Stop_Production,FALSE)	
16		= SET.VALUE(Begin_Production_Level,MASTER.XLM!Factory_Start)	
17		= SET.VALUE(Stop_Production_Level,MASTER.XLM!Factory_Stop)	
18		= SET.VALUE(Production_Rate,MASTER.XLM!Factory_Rate)	
19		= SET.VALUE(Holding_Cost,MASTER.XLM!Factory_HC)	
20		= SET.VALUE(Setup_Cost,MASTER.XLM!Factory_PC)	
21		= SET.VALUE(Shortage_Cost,MASTER.XLM!Factory_SC)	
22		= SET.VALUE(Day,0)	
23		= SET.VALUE(Back_Order,0)	
24		= SET.VALUE(Requested_Order,0)	
25		= SET.VALUE(Production,FALSE)	
26		= SET.VALUE(Setup,FALSE)	
27		= SET.VALUE(Begin_Production,FALSE)	
28		= SET.VALUE(Stop_Production,FALSE)	
29		= SET.VALUE(Total_Cost,0)	
30		= SET.VALUE(Total_Holding_Cost,0)	
31		= SET.VALUE(Total_Setup_Cost,0)	
32		= SET.VALUE(Total_Shortage_Cost,0)	
33			
34		= IF(MASTER.XLM!Prompt_Individual=TRUE)	User Choose Variables
35		= DIALOG.BOX(Input_Box)	
36		= IF(B35=FALSE)	End Simulation by user
37	QUIT	= ALERT("Simulation Cancelled",3)	
38		= MESSAGE(FALSE)	
39		= ACTIVATE("MASTER.XLS")	
40		= HALT()	
41		= END.IF()	
42		= END.IF()	
43			
44			
45		= SET.VALUE(Current_Inventory,Beginning_Inventory)	Set Current Inventory
46			
47		= SET.NAME("Output_Data",$N$4 OFFSET($N$4,(MASTER.XLM!Total_Iterations+1),9))	Clear Output Data
48		= FORMULA.GOTO(Output_Data,$N$4)	
49		= CLEAR(3)	
50		= VSCROLL(1,TRUE)	
51		= RUN(FACTORY.XLM!Factory_Output)	Record Initial Variables
52		=END.IF()	
53			
54			
55			**Begin Routine**
56			
57	Day	=MASTER.XLM!Day	Counter for Reference
58			
59		=Current_Inventory-Back_Order	Fill Backorder if Possible
60		=IF(B59>=0,0,ABS(B59))	Determine Size of Backorder
61		=IF(B59>0,B59,0)	Ensure Inventory not Negative
62		=Current_Inventory-B61	Determine Change in Inventory
63			
64		=IF(MASTER.XLM!Warehouse_Number=1)	Fill Demand if Possible
65	Order_1	= B61-'WARE-1.XLM'!Order	from relevant warehouse
66		= SET.VALUE(Requested_Order,Order_1)	
67		=ELSE.IF(MASTER.XLM!Warehouse_Number=2)	
68	Order_2	= B61-'WARE-2.XLM'!Order	
69		= SET.VALUE(Requested_Order,Order_2)	

	A	B	C
1	*names*	**Commands**	*comments*
2			
70		=ELSE()	
71	Order_3	= B61-'WARE-3.XLM'!Order	
72		= SET.VALUE(Requested_Order,Order_3)	
73		=END.IF()	
74			
75	Requested_Order	=Requested_Order	
76		=IF(Requested_Order>=0,Requested_Order,0)	Ensure Inventory not Negative
77		=B61-B76	Determine Change in Inventory
78	Shipment	=B62+B77	Determine Amount Shipped
79	Back_Order	=IF(Requested_Order<0,B60+ABS(Requested_Order),B60)	Adjust Backorder Level
80			
81	Current_Inventory	=B76+IF(Production=TRUE,Production_Rate,0)	Receive Yesterday's Production
82			
83		=IF(Production=FALSE)	Should Production Begin?
84		=IF(MASTER.XLM!P_Start_Iteration=TRUE,GOTO(B103),)	Skip Start Production ? if Startup
85			
86		= IF(Prompt_Begin_Production=TRUE)	Start Production ?
87		= SET.VALUE(K102,Day)	
88		= IF(MASTER.XLM!Warehouse_Number=3)	
89		= SET.VALUE(K104,'WARE-3.XLM'!Order)	
90		= ELSE.IF(MASTER.XLM!Warehouse_Number=2)	
91		= SET.VALUE(K104,'WARE-2.XLM'!Order)	
92		= ELSE()	
93		= SET.VALUE(K104,'WARE-1.XLM'!Order)	
94		= END.IF()	
95		= SET.VALUE(K99,Current_Inventory)	
96		= SET.VALUE(Begin_Production,FALSE)	
97		= DIALOG.BOX(Start_Box)	
98		= IF(B97=FALSE,SET.VALUE(Prompt_Begin_Production,FALSE),)	
99			
100	Production	= IF(Begin_Production=TRUE,TRUE,FALSE)	Production Flag
101	Setup	= IF(Production=TRUE,TRUE,FALSE)	Setup Flag
102		= ELSE.IF(Prompt_Begin_Production=FALSE)	
103		= IF(Current_Inventory<Begin_Production_Level)	
104		= SET.VALUE(Production,TRUE)	
105		= SET.VALUE(Setup,TRUE)	Setup Flag
106		= END.IF()	
107		= IF(MASTER.XLM!P_Start_Iteration=TRUE,GOTO(B143),)	
108		= END.IF()	
109			
110		=ELSE.IF(Production=TRUE)	Stop Production?
111		=IF(MASTER.XLM!P_Start_Iteration=TRUE,GOTO(B130),)	Skip Stop Production ? if Startup
112			
113		= IF(Prompt_Stop_Production=TRUE)	
114		= SET.VALUE(K132,Day)	
115		= IF(MASTER.XLM!Warehouse_Number=3)	
116		= SET.VALUE(K134,'WARE-3.XLM'!Order)	
117		= ELSE.IF(MASTER.XLM!Warehouse_Number=2)	
118		= SET.VALUE(K134,'WARE-2.XLM'!Order)	
119		= ELSE()	
120		= SET.VALUE(K134,'WARE-1.XLM'!Order)	
121		= END.IF()	
122		= SET.VALUE(K129,Current_Inventory-Back_Order)	
123		= SET.VALUE(Stop_Production,FALSE)	
124		= DIALOG.BOX(Stop_Box)	
125		= IF(B124=FALSE,SET.VALUE(Prompt_Stop_Production,FALSE),)	
126		= IF(Stop_Production=TRUE)	
127		= SET.VALUE(Production,FALSE)	
128		= END.IF()	
129		= ELSE.IF(Prompt_Stop_Production=FALSE)	
130		= IF(Current_Inventory>Stop_Production_Level)	
131		= SET.VALUE(Production,FALSE)	
132		= END.IF()	
133		= IF(MASTER.XLM!P_Start_Iteration=TRUE,GOTO(B143),)	Skip Cost if Startup
134		= END.IF()	
135		=END.IF()	
136			
137	Total_Shortage_Cost	=Back_Order*Shortage_Cost+B137	Determine Costs
138	Total_Holding_Cost	=Current_Inventory*Holding_Cost+B138	
139	Total_Setup_Cost	=Setup*Setup_Cost+B139	
140	Total_Cost	=B137+B138+B139	

	A	B	C
1	*names*	**Commands**	*comments*
2			
141		=SET.VALUE(Setup,FALSE)	Clear Setup Flag
142			
143		=RUN(FACTORY.XLM!Factory_Output)	Record Output
144			
145		=RETURN(Shipment)	
146			
147			
148			**Output Routine**
149			
150	Factory_Output	=FORMULA(Day,OFFSET($N$4,Day,0))	**Day**
151		=FORMULA(Current_Inventory,OFFSET($N$4,Day,1))	**Inventory**
152		=FORMULA(Back_Order,OFFSET($N$4,Day,2))	**Back_Order**
153		=FORMULA(Total_Shortage_Cost,OFFSET($N$4,Day,3))	**Total_Shortage_Cost**
154		=FORMULA(Total_Holding_Cost,OFFSET($N$4,Day,4))	**Total_Holding_Cost**
155		=FORMULA(Total_Setup_Cost,OFFSET($N$4,Day,5))	**Total_Setup_Cost**
156		=FORMULA(Total_Cost,OFFSET($N$4,Day,6))	**Total_Cost**
157		=FORMULA(Requested_Order,OFFSET($N$4,Day,7))	**Demand**
158		=FORMULA(Production,OFFSET($N$4,Day,8))	**Production**
159		=FORMULA(Shipment,OFFSET($N$4,Day,9))	**Shipment**
160			
161		=RETURN()	

	A	B	C
1	*names*	**Commands**	*comments*
2			
70		=ELSE()	
71	Order_3	= B61-'WARE-3.XLM'!Order	
72		= SET.VALUE(Requested_Order,Order_3)	
73		=END.IF()	
74			
75	Requested_Order	=Requested_Order	
76		=IF(Requested_Order>=0,Requested_Order,0)	Ensure Inventory not Negative
77		=B61-B76	Determine Change in Inventory
78	Shipment	=B62+B77	Determine Amount Shipped
79	Back_Order	=IF(Requested_Order<0,B60+ABS(Requested_Order),B60)	Adjust Backorder Level
80			
81	Current_Inventory	=B76+IF(Production=TRUE,Production_Rate,0)	Receive Yesterday's Production
82			
83		=IF(Production=FALSE)	Should Production Begin?
84		=IF(MASTER.XLM!P_Start_Iteration=TRUE,GOTO(B103),)	Skip Start Production ? if Startup
85			
86		= IF(Prompt_Begin_Production=TRUE)	Start Production ?
87		= SET.VALUE(K102,Day)	
88		= IF(MASTER.XLM!Warehouse_Number=3)	
89		= SET.VALUE(K104,'WARE-3.XLM'!Order)	
90		= ELSE.IF(MASTER.XLM!Warehouse_Number=2)	
91		= SET.VALUE(K104,'WARE-2.XLM'!Order)	
92		= ELSE()	
93		= SET.VALUE(K104,'WARE-1.XLM'!Order)	
94		= END.IF()	
95		= SET.VALUE(K99,Current_Inventory)	
96		= SET.VALUE(Begin_Production,FALSE)	
97		= DIALOG.BOX(Start_Box)	
98		= IF(B97=FALSE,SET.VALUE(Prompt_Begin_Production,FALSE),)	
99			
100	Production	= IF(Begin_Production=TRUE,TRUE,FALSE)	Production Flag
101	Setup	= IF(Production=TRUE,TRUE,FALSE)	Setup Flag
102		= ELSE.IF(Prompt_Begin_Production=FALSE)	
103		= IF(Current_Inventory<Begin_Production_Level)	
104		= SET.VALUE(Production,TRUE)	
105		= SET.VALUE(Setup,TRUE)	Setup Flag
106		= END.IF()	
107		= IF(MASTER.XLM!P_Start_Iteration=TRUE,GOTO(B143),)	
108		= END.IF()	
109			
110		=ELSE.IF(Production=TRUE)	Stop Production?
111		=IF(MASTER.XLM!P_Start_Iteration=TRUE,GOTO(B130),)	Skip Stop Production ? if Startup
112			
113		= IF(Prompt_Stop_Production=TRUE)	
114		= SET.VALUE(K132,Day)	
115		= IF(MASTER.XLM!Warehouse_Number=3)	
116		= SET.VALUE(K134,'WARE-3.XLM'!Order)	
117		= ELSE.IF(MASTER.XLM!Warehouse_Number=2)	
118		= SET.VALUE(K134,'WARE-2.XLM'!Order)	
119		= ELSE()	
120		= SET.VALUE(K134,'WARE-1.XLM'!Order)	
121		= END.IF()	
122		= SET.VALUE(K129,Current_Inventory-Back_Order)	
123		= SET.VALUE(Stop_Production,FALSE)	
124		= DIALOG.BOX(Stop_Box)	
125		= IF(B124=FALSE,SET.VALUE(Prompt_Stop_Production,FALSE),)	
126		= IF(Stop_Production=TRUE)	
127		= SET.VALUE(Production,FALSE)	
128		= END.IF()	
129		= ELSE.IF(Prompt_Stop_Production=FALSE)	
130		= IF(Current_Inventory>Stop_Production_Level)	
131		= SET.VALUE(Production,FALSE)	
132		= END.IF()	
133		= IF(MASTER.XLM!P_Start_Iteration=TRUE,GOTO(B143),)	Skip Cost if Startup
134		= END.IF()	
135		=END.IF()	
136			
137	Total_Shortage_Cost	=Back_Order*Shortage_Cost+B137	Determine Costs
138	Total_Holding_Cost	=Current_Inventory*Holding_Cost+B138	
139	Total_Setup_Cost	=Setup*Setup_Cost+B139	
140	Total_Cost	=B137+B138+B139	

	A	B	C
1	*names*	**Commands**	comments
2			
141		=SET.VALUE(Setup,FALSE)	Clear Setup Flag
142			
143		=RUN(FACTORY.XLM!Factory_Output)	Record Output
144			
145		=*RETURN(Shipment)*	
146			
147			
148			**Output Routine**
149			
150	Factory_Output	=FORMULA(Day,OFFSET($N$4,Day,0))	**Day**
151		=FORMULA(Current_Inventory,OFFSET($N$4,Day,1))	**Inventory**
152		=FORMULA(Back_Order,OFFSET($N$4,Day,2))	**Back_Order**
153		=FORMULA(Total_Shortage_Cost,OFFSET($N$4,Day,3))	**Total_Shortage_Cost**
154		=FORMULA(Total_Holding_Cost,OFFSET($N$4,Day,4))	**Total_Holding_Cost**
155		=FORMULA(Total_Setup_Cost,OFFSET($N$4,Day,5))	**Total_Setup_Cost**
156		=FORMULA(Total_Cost,OFFSET($N$4,Day,6))	**Total_Cost**
157		=FORMULA(Requested_Order,OFFSET($N$4,Day,7))	**Demand**
158		=FORMULA(Production,OFFSET($N$4,Day,8))	**Production**
159		=FORMULA(Shipment,OFFSET($N$4,Day,9))	**Shipment**
160			
161		=RETURN()	

APPENDIX B
QUEUING WORKSHEET AND MACRO

Queueing Simulation Model

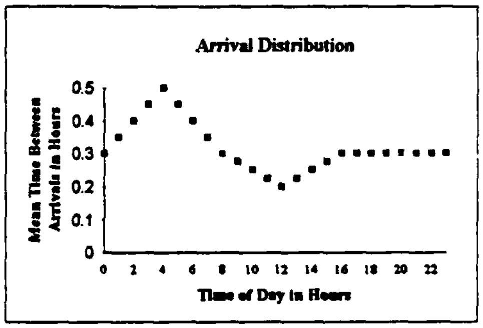
Perform Simulation

Initial Data (Do not change on Worksheet)

Patient

Type	Description	Frequency
1	Open Wounds	8
2	Closed Injuries	13
3	Multiple Trauma	33
4	Visceral Complaints	20
5	Chronic Complaints	26

Number of Beds 5



Nurse

Nurses Available: 2

	Shift Times			Normal Distribution		
	On	Off	Type	Lower	Upper	Standard Deviation
#1	00:00	24:00	1	0.00	0.00	1.00
#2	08:00	16:00	2	0.20	0.30	1.00
#3	00:00	00:00	3	0.15	0.25	1.00
#4	00:00	00:00	4	0.10	0.20	1.00
			5	0.05	0.15	1.00

Doctor

Doctors Available: 2

	Shift Times			Normal Distribution		
	On	Off	Type	Lower	Upper	Standard Deviation
#1	00:00	24:00	1	0.25	0.35	1.00
#2	08:00	16:00	2	0.20	0.30	1.00
#3	00:00	00:00	3	0.15	0.25	1.00
#4	00:00	00:00	4	0.10	0.20	1.00
			5	0.05	0.15	1.00

QUEUING MACRO

	A	B	C
1	*names*	Commands	*comments*
2			
3		Summary Information	
4	*Title:*	Queuing Macro	
5	*Version:*	v1.0	
6	*Author*	Dixon Hicks	
7	*Corporation:*	Naval Postgraduate School	
8	*Creation Date:*	Friday, February 5, 1993	
9			
10	*CommandWindow*	CommandWindow	
11	QUEUE		
12		=SET.VALUE(Master_Clock_Time,0)	Set Initial Variable
13		=SET.VALUE(Active_Clock,1)	
14		=SET.VALUE(Unit_Number,1)	
15		=SET.VALUE(Bed_Queue,0)	
16		=SET.VALUE(Active_Pointer,1)	
17		=SET.VALUE(Beds_Occupied,0)	
18		=SET.VALUE(Registration_Queue,0)	
19		=SET.VALUE(Total_Patients,0)	
20		=SET.VALUE(Nurses_Occupied,0)	
21		=SET.VALUE(Doctors_Occupied,0)	
22		=SET.VALUE(Total_Type_1,0)	
23		=SET.VALUE(Total_Type_2,0)	
24		=SET.VALUE(Total_Type_3,0)	
25		=SET.VALUE(Total_Type_4,0)	
26		=SET.VALUE(Total_Type_5,0)	
27		=SET.VALUE(Active_Nurse,2)	
28		=SET.VALUE(Active_Doctor,6)	
29		=SET.VALUE(End_Simulation_Time,End_Simulation_Time/60)	
30		=SET.VALUE(Bed_Queue_First,TRUE)	
31		=SET.VALUE(Registration_Queue_First,TRUE)	
32		=SET.VALUE(System_Counter,1)	
33		=SET.VALUE(Reg_Queue_Counter,1)	
34		=SET.VALUE(Bed_Queue_Counter,1)	
35		=SET.VALUE(Nurse_1_Total_1,0)	
36		=SET.VALUE(Nurse_1_Total_2,0)	
37		=SET.VALUE(Nurse_1_Total_3,0)	
38		=SET.VALUE(Nurse_1_Total_4,0)	
39		=SET.VALUE(Nurse_1_Total_5,0)	
40		=SET.VALUE(Nurse_2_Total_1,0)	
41		=SET.VALUE(Nurse_2_Total_2,0)	
42		=SET.VALUE(Nurse_2_Total_3,0)	
43		=SET.VALUE(Nurse_2_Total_4,0)	
44		=SET.VALUE(Nurse_2_Total_5,0)	
45		=SET.VALUE(Nurse_3_Total_1,0)	
46		=SET.VALUE(Nurse_3_Total_2,0)	
47		=SET.VALUE(Nurse_3_Total_3,0)	
48		=SET.VALUE(Nurse_3_Total_4,0)	
49		=SET.VALUE(Nurse_3_Total_5,0)	
50		=SET.VALUE(Nurse_4_Total_1,0)	
51		=SET.VALUE(Nurse_4_Total_2,0)	
52		=SET.VALUE(Nurse_4_Total_3,0)	
53		=SET.VALUE(Nurse_4_Total_4,0)	
54		=SET.VALUE(Nurse_4_Total_5,0)	
55		=SET.VALUE(Doctor_1_Total_1,0)	
56		=SET.VALUE(Doctor_1_Total_2,0)	
57		=SET.VALUE(Doctor_1_Total_3,0)	
58		=SET.VALUE(Doctor_1_Total_4,0)	
59		=SET.VALUE(Doctor_1_Total_5,0)	
60		=SET.VALUE(Doctor_2_Total_1,0)	
61		=SET.VALUE(Doctor_2_Total_2,0)	
62		=SET.VALUE(Doctor_2_Total_3,0)	
63		=SET.VALUE(Doctor_2_Total_4,0)	
64		=SET.VALUE(Doctor_2_Total_5,0)	
65		=SET.VALUE(Doctor_3_Total_1,0)	
66		=SET.VALUE(Doctor_3_Total_2,0)	
67		=SET.VALUE(Doctor_3_Total_3,0)	
68		=SET.VALUE(Doctor_3_Total_4,0)	
69		=SET.VALUE(Doctor_3_Total_5,0)	
70		=SET.VALUE(Doctor_4_Total_1,0)	
71		=SET.VALUE(Doctor_4_Total_2,0)	
72		=SET.VALUE(Doctor_4_Total_3,0)	
73		=SET.VALUE(Doctor_4_Total_4,0)	
74		=SET.VALUE(Doctor_4_Total_5,0)	
75		=SET.VALUE(Bed_1_Total,0)	
76		=SET.VALUE(Bed_2_Total,0)	
77		=SET.VALUE(Bed_3_Total,0)	
78		=SET.VALUE(Bed_4_Total,0)	
79		=SET.VALUE(Bed_5_Total,0)	
80		=SET.VALUE(Bed_6_Total,0)	
81		=SET.VALUE(Bed_7_Total,0)	
82		=SET.VALUE(Bed_8_Total,0)	
83		=SET.VALUE(Bed_9_Total,0)	
84		=SET.VALUE(Bed_10_Total,0)	
85			
86		=DIALOG.BOX(INPUT_BOX_1)	User Choose Variables
87		=IF(B86=FALSE,GOTO(QUIT))	

	A	B	C
1	names	Commands	comments
2			
88		=DIALOG.BOX(INPUT_BOX_2)	
89		=IF(B88=FALSE,GOTO(QUIT))	
90		=DIALOG.BOX(INPUT_BOX_3)	
91		=IF(B90=FALSE,GOTO(QUIT))	
92		=DIALOG.BOX(INPUT_BOX_4)	
93		=IF(B92=FALSE)	
94	QUIT	= ALERT("Simulation Cancelled",3)	End Simulation by user
95		= MESSAGE(FALSE)	
96		= ACTIVATE("Q-MASTER.XLS")	
97		= HALT()	
98		=END.IF()	
99		=ECHO(FALSE)	
100		=SET.VALUE(End_Simulation_Time,End_Simulation_Time*60)	
101			
102		=FORMULA.GOTO(Start_Box,TRUE)	Clear Matrix Boxes
103		=CLEAR(3)	
104		= FORMULA.GOTO(Unit_Box,FALSE)	
105		=CLEAR(3)	
106		=SET.VALUE(Time_Till_Next_Event,End_Simulation_Time*2)	
107		=FORMULA.GOTO(Next_Event_Box,FALSE)	
108		=CLEAR(3)	
109		=FORMULA.GOTO(AJ3:AN980,FALSE)	
110		=CLEAR(3)	
111		=FORMULA.GOTO(AP3:AT980,FALSE)	
112		=CLEAR(3)	
113		=FORMULA.GOTO(AV3:AZ980,FALSE)	
114		=CLEAR(3)	
115		=FORMULA.GOTO(Output_Box_2,FALSE)	
116		=CLEAR(3)	
117		=FORMULA.GOTO(Output_Box_1,TRUE)	
118		=CLEAR(3)	
119		=VSCROLL(1,TRUE)	
120		=ACTIVATE.PREV()	
121		=SET.VALUE(OFFSET(Next_Event_Reference,1,1),1)	Place Unit #1 as First Unit
122		=SET.VALUE(OFFSET(Unit_Reference,2,1),1)	Place Unit #1 in System Entry
123		=SET.VALUE(OFFSET(Unit_Reference,1,1),Master_Clock_Time+RUN(Arrival_Distribution))	Retrieve First Arrival Time
124		=FORMULA(System_Counter,OFFSET(AJ2,System_Counter,0))	
125		=FORMULA(0,OFFSET(AJ2,System_Counter,1))	
126		=FORMULA(OFFSET(Unit_Reference,1,1),OFFSET(AJ2,System_Counter,2))	
127			
128			Begin Routine
129	START	=IF(Master_Clock_Time>End_Simulation_Time)	
130		= RUN(END)	Determine if Simulation is Complete
131		= ALERT("Simulation Complete",3)	
132		= MESSAGE(FALSE)	
133		= ACTIVATE("Q-MASTER.XLS")	
134		= RETURN()	
135		=ELSE.IF(Unit_Number>Maximum_Units)	
136		= RUN(END)	
137		= ALERT("Simulation Complete",3)	
138		= MESSAGE(FALSE)	
139		= ACTIVATE("Q-MASTER.XLS")	
140		= RETURN()	
141		=END.IF()	
142			
143	Minimum_Time	=MIN(Time_Till_Next_Event)	Find Lowest Time till next event
144	Master_Clock_Time	=Master_Clock_Time+Minimum_Time	Adjust Master Clock
145	Master_Clock_24Hr	=Master_Clock_Time-1440*INT(Master_Clock_Time/1440)	Determine 24Hr Equivalent Time
146			
147	Active_Clock	=MATCH(Minimum_Time,Time_Till_Next_Event,0)	Find Active Parameters
148	Active_Pointer	=OFFSET(Unit_Reference,2,Active_Clock)	
149			
150		=FORMULA((System_Entry-Minimum_Time),System_Entry)	Subtract Minimum Time from all
151		=FORMULA((Nurse_1_Clock-Minimum_Time),Nurse_1_Clock)	
152		=FORMULA((Nurse_2_Clock-Minimum_Time),Nurse_2_Clock)	
153		=FORMULA((Nurse_3_Clock-Minimum_Time),Nurse_3_Clock)	
154		=FORMULA((Nurse_4_Clock-Minimum_Time),Nurse_4_Clock)	
155		=FORMULA((Doctor_1_Clock-Minimum_Time),Doctor_1_Clock)	
156		=FORMULA((Doctor_2_Clock-Minimum_Time),Doctor_2_Clock)	
157		=FORMULA((Doctor_3_Clock-Minimum_Time),Doctor_3_Clock)	
158		=FORMULA((Doctor_4_Clock-Minimum_Time),Doctor_4_Clock)	
159			
160	Doctor_Available	=0	
161		=IF(Doctor_Number>1)	Determine if Doctor On/Off Shift
162		= IF(W148>W153,W148-2400,W148)	
163		= IF(Master_Clock_24Hr>INT(B162/100)*60+MOD(B162,100),SET.VALUE(Doctor_Available,Doctor_Available+1),)	
164		= IF(W149>W154,W149-2400,W149)	
165		= IF(Master_Clock_24Hr>INT(B164/100)*60+MOD(B164,100),SET.VALUE(Doctor_Available,Doctor_Available+1),)	
166		= IF(Master_Clock_24Hr>INT(W153/100)*60+MOD(W153,100),SET.VALUE(Doctor_Available,Doctor_Available-1),)	
167		= IF(Master_Clock_24Hr>INT(W154/100)*60+MOD(W154,100),SET.VALUE(Doctor_Available,Doctor_Available-1),)	
168		=END.IF()	
169		=IF(Doctor_Number>2)	
170		= IF(W150>W155,W150-2400,W150)	
171		= IF(Master_Clock_24Hr>INT(B170/100)*60+MOD(B170,100),SET.VALUE(Doctor_Available,Doctor_Available+1),)	
172		= IF(Master_Clock_24Hr>INT(W155/100)*60+MOD(W155,100),SET.VALUE(Doctor_Available,Doctor_Available-1),)	
173		=END.IF()	
174		=IF(Doctor_Number>3)	
175		= IF(W151>W156,W151-2400,W151)	
176		= IF(Master_Clock_24Hr>INT(B175/100)*60+MOD(B175,100),SET.VALUE(Doctor_Available,Doctor_Available+1),)	
177		= IF(Master_Clock_24Hr>INT(W156/100)*60+MOD(W156,100),SET.VALUE(Doctor_Available,Doctor_Available-1),)	
178		=END.IF()	

	A	B	C
1	names	Commands	comments
2			
179		=SET.NAME("Doctor_Box",$K$4:OFFSET($K$4,0,Doctor_Available-1))	Set Doctor available Box
180			
181	Nurse_Available	=0	
182		=IF(Nurse_Number>1)	Determine if Nurse On/Off Shift
183		= IF(W101>W106,W101-2400,W101)	
184		= IF(Master_Clock_24Hr>INT(B183/100)*60+MOD(B183,100),SET.VALUE(Nurse_Available,Nurse_Available+1),)	
185		= IF(W102>W107,W102-2400,W102)	
186		= IF(Master_Clock_24Hr>INT(B185/100)*60+MOD(B185,100),SET.VALUE(Nurse_Available,Nurse_Available+1),)	
187		= IF(Master_Clock_24Hr>INT(W106/100)*60+MOD(W106,100),SET.VALUE(Nurse_Available,Nurse_Available-1),)	
188		= IF(Master_Clock_24Hr>INT(W107/100)*60+MOD(W107,100),SET.VALUE(Nurse_Available,Nurse_Available-1),)	
189		=END.IF()	
190		=IF(Nurse_Number>2)	
191		= IF(W103>W108,W103-2400,W103)	
192		= IF(Master_Clock_24Hr>INT(B191/100)*60+MOD(B191,100),SET.VALUE(Nurse_Available,Nurse_Available+1),)	
193		= IF(Master_Clock_24Hr>INT(W108/100)*60+MOD(W108,100),SET.VALUE(Nurse_Available,Nurse_Available-1),)	
194		=END.IF()	
195		=IF(Nurse_Number>3)	
196		= IF(W104>W109,W104-2400,W104)	
197		= IF(Master_Clock_24Hr>INT(B196/100)*60+MOD(B196,100),SET.VALUE(Nurse_Available,Nurse_Available+1),)	
198		= IF(Master_Clock_24Hr>INT(W109/100)*60+MOD(W109,100),SET.VALUE(Nurse_Available,Nurse_Available-1),)	
199		=END.IF()	
200		=SET.NAME("Nurse_Box",$G$4:OFFSET($G$4,0,Nurse_Available-1))	Set Nurse available Box
201			
202		=IF(Active_Clock=1)	Perform System Entry Steps
203		= RUN(Arrival)	
204			
205		=ELSE.IF(Active_Clock<6)	Perform Nurse Complete Steps
206		= SET.VALUE(Active_Nurse,Active_Clock)	
207		= RUN(Nurse_A)	Free up Nurse
208		= RUN(Bed_Routine_B)	See if Bed available
209		= IF(Beds_Occupied-Doctors_Occupied>0,RUN(Doctor_B))	See if Doctor available
210		= IF(Registration_Queue>0,RUN(Nurse_B))	See if Nurse Needed
211			
212		=ELSE()	Perform Doctor Complete Steps
213		= SET.VALUE(Active_Doctor,Active_Clock)	
214		= RUN(Doctor_A)	Free up Doctor
215		= RUN(Bed_Routine_A)	Clear Bed
216		= RUN(Patient_Complete)	Transfer Data to Output
217		= IF(Bed_Queue>0,RUN(Bed_Routine_B))	See if Bed available
218		= IF(Beds_Occupied-Doctors_Occupied>0,RUN(Doctor_B))	See if Doctor available
219			
220		=END.IF()	
221		=GOTO(START)	
222			
223			Arrival Routine
224			
225	Arrival		
226	Total_Patients	=Total_Patients+1	Gain Another Patient
227		=MESSAGE(TRUE,Total_Patients)	Display Number of Patients
228			
229	Unit_Number	= Unit_Number+1	Add One to Unit Reference
230		= SET.VALUE(System_Entry,RUN(Arrival_Distribution))	Determine Next Arrival Time
231			
232		= SET.VALUE(Blank_Counter,1)	Find A Blank Row for Next Event
233		= WHILE(ISBLANK(OFFSET(Next_Event_Reference,Blank_Counter,1))=FALSE)	
234	Blank_Counter	= Blank_Counter+1	
235		= IF(Blank_Counter=76)	
236		= ALERT("Excess Queuing, Model Failure",3)	
237		= MESSAGE(FALSE)	
238		= ACTIVATE("MASTER.XLS")	
239		= HALT()	
240		= END.IF()	
241		= NEXT()	
242		= SET.VALUE(OFFSET(Next_Event_Reference,Blank_Counter,1),Unit_Number)	Place Next Unit # in Blank Row
243		= FORMULA(Blank_Counter,OFFSET(Unit_Reference,2,1))	Place Pointer in Unit Box
244		= FORMULA(RUN(Patient_Category),OFFSET(Next_Event_Reference,Active_Pointer,2))	Assign Patient Type
245		= FORMULA(Master_Clock_Time,OFFSET(Next_Event_Reference,Active_Pointer,3))	Mark Time in System
246			
247		= FORMULA(TRUE,OFFSET(Next_Event_Reference,Active_Pointer,5))	Time Flags
248		= FORMULA(TRUE,OFFSET(Next_Event_Reference,Active_Pointer,6))	
249		= FORMULA(TRUE,OFFSET(Next_Event_Reference,Active_Pointer,7))	
250		= FORMULA(TRUE,OFFSET(Next_Event_Reference,Active_Pointer,8))	
251		= FORMULA(TRUE,OFFSET(Next_Event_Reference,Active_Pointer,9))	
252			
253		= IF(Patient_Type=1)	If Cate 1, Send to Bed Queue
254	Bed_Queue	= Bed_Queue+1	Add One to Bed Queue
255			
256		= IF(Bed_Queue_First=TRUE)	Tabulate Bed Queue Statistics
257		= FORMULA(Bed_Queue_Counter,OFFSET(AV2,Bed_Queue_Counter,0))	
258		= FORMULA(Bed_Queue-1,OFFSET(AV2,Bed_Queue_Counter,1))	
259		= FORMULA(Master_Clock_Time,OFFSET(AV2,Bed_Queue_Counter,2))	
260	Bed_Queue_First	= SET.VALUE(Bed_Queue_First,FALSE)	
261		= END.IF()	
262	Bed_Queue_Counter	= Bed_Queue_Counter+1	
263		= FORMULA(Bed_Queue_Counter,OFFSET(AV2,Bed_Queue_Counter,0))	
264		= FORMULA(Bed_Queue,OFFSET(AV2,Bed_Queue_Counter,1))	
265		= FORMULA(Master_Clock_Time,OFFSET(AV2,Bed_Queue_Counter,2))	
266		= FORMULA(Master_Clock_Time-OFFSET(AV2,Bed_Queue_Counter-1,2),OFFSET(AV2,Bed_Queue_Counter-1,3))	
267		= FORMULA((OFFSET(AV2,Bed_Queue_Counter-1,1)*OFFSET(AV2,Bed_Queue_Counter-1,3)),OFFSET(AV2,Bed_Queue	
268		= FORMULA(Master_Clock_Time,OFFSET(Next_Event_Reference,Active_Pointer,7))	Mark Time in Bed Queue
269		= FORMULA(3,OFFSET(Next_Event_Reference,Active_Pointer,4))	Make Unit State 3

	A	3	C
1	names	Commands	comments
2			
270		= RUN(Bed_Routine_B)	See if Bed Available
271		= RUN(Doctor_B)	See if Doctor Available
272			
273		= ELSE.IF(Patient_Type<>1)	If Not Case 1, Register
274	Registration_Queue	= Registration_Queue+1	Add to Registration Queue
275			
276		= IF(Registration_Queue_First=TRUE)	Tabulate Registration Queue Statisti
277		= FORMULA(Reg_Queue_Counter,OFFSET(AP2,Reg_Queue_Counter,0))	
278		= FORMULA(Registration_Queue-1,OFFSET(AP2,Reg_Queue_Counter,1))	
279		= FORMULA(Master_Clock_Time,OFFSET(AP2,Reg_Queue_Counter,2))	
280	Registration_Queue_Fi	= SET.VALUE(Registration_Queue_First,FALSE)	
281		= END.IF()	
282	Reg_Queue_Counter	= Reg_Queue_Counter+1	
283		= FORMULA(Reg_Queue_Counter,OFFSET(AP2,Reg_Queue_Counter,0))	
284		= FORMULA(Registration_Queue,OFFSET(AP2,Reg_Queue_Counter,1))	
285		= FORMULA(Master_Clock_Time,OFFSET(AP2,Reg_Queue_Counter,2))	
286		= FORMULA(Master_Clock_Time-OFFSET(AP2,Reg_Queue_Counter-1,2),OFFSET(AP2,Reg_Queue_Counter-1,3))	
287		= FORMULA((OFFSET(AP2,Reg_Queue_Counter-1,1)*OFFSET(AP2,Reg_Queue_Counter-1,3)),OFFSET(AP2,Reg_Queue_	
288		= FORMULA(Master_Clock_Time,OFFSET(Next_Event_Reference,Active_Pointer,5))	Mark Time in Registration Queue
289		= FORMULA(1,OFFSET(Next_Event_Reference,Active_Pointer,4))	Make Unit State 1
290		= RUN(Nurse_B)	See if Nurse Available
291		= END.IF()	
292			
293	System_Counter	=System_Counter+1	Tabulate System Statistics
294		=FORMULA(System_Counter,OFFSET(AJ2,System_Counter,0))	
295		=FORMULA(Registration_Queue+Nurses_Occupied+Bed_Queue+Beds_Occupied,OFFSET(AJ2,System_Counter,1))	
296		=FORMULA(Master_Clock_Time,OFFSET(AJ2,System_Counter,2))	
297		=FORMULA(Master_Clock_Time-OFFSET(AJ2,System_Counter-1,2),OFFSET(AJ2,System_Counter-1,3))	
298		=FORMULA((OFFSET(AJ2,System_Counter-1,1)*OFFSET(AJ2,System_Counter-1,3)),OFFSET(AJ2,System_Counter-1,4))	
299			
300		=RETURN()	
301			
302			Arrival Distribution Routine
303			
304			
305	Arrival_Distribution	=RAND()	
306			
307		=VLOOKUP(Master_Clock_24Hr/60,Lambda_Box,1)	Calculate Lambda
308		=VLOOKUP((Master_Clock_24Hr+60)/60,Lambda_Box,1)	
309		=VLOOKUP(Master_Clock_24Hr/60,Lambda_Box,2)	
310		=VLOOKUP((Master_Clock_24Hr+60)/60,Lambda_Box,2)	
311	Lambda	=(1/(((Master_Clock_24Hr/60-B307)/(B308-B307))*(B310-B309)+B309))/60	
312			
313		=SET.VALUE(Count_Lambda,0)	
314		=FOR("Counter",0,240)	
315	Count_Lambda	= Count_Lambda+1	
316	Hourly_Arrivals	= IF($B$305>EXPONDIST(Count_Lambda,Lambda,TRUE),Count_Lambda+1,Count_Lambda)	
317		= IF(Hourly_Arrivals=Count_Lambda,RETURN(Hourly_Arrivals),)	
318		=NEXT()	
319			
320			Patient_Category Routine
321			
322	Patient_Category	=RAND()	
323		=SUM(W81:W88)	Determine Patient Type
324		=Patient_Freq_1/B323	
325		=Patient_Freq_2/B323	
326		=Patient_Freq_3/B323	
327		=Patient_Freq_4/B323	
328		=IF($B$322>B324,2,1)	
329		=IF($B$322>B324+B325,3,1)	
330		=IF($B$322>B324+B325+B326,4,1)	
331		=IF($B$322>B324+B325+B326+B327,5,1)	
332	Patient_Type	=MAX(B328:B331)	
333			
334	Total_Type_1	=IF(Patient_Type=1,Total_Type_1+1,Total_Type_1)	Tally Patients
335	Total_Type_2	=IF(Patient_Type=2,Total_Type_2+1,Total_Type_2)	
336	Total_Type_3	=IF(Patient_Type=3,Total_Type_3+1,Total_Type_3)	
337	Total_Type_4	=IF(Patient_Type=4,Total_Type_4+1,Total_Type_4)	
338	Total_Type_5	=IF(Patient_Type=5,Total_Type_5+1,Total_Type_5)	
339			
340		=RETURN(Patient_Type)	
341			
342			Bed Ro[illegible]
343			
344	Bed_Routine_A		Remove Patient from Bed
345	2	=FORMULA(OFFSET(Next_Event_Reference,Active_Pointer,10),A345)	Get Occupied Bed Number
346	10048.1345564626	=FORMULA(OFFSET(Start_Reference,A345,1),A346)	Retrieve when Bed Occupied
347		=Master_Clock_Time-A346	Time Bed was Occupied
348			
349	Bed_1_Total	=IF(A345=1,Bed_1_Total+B$347,Bed_1_Total)	Tally Bed Occupied Time
350	Bed_2_Total	=IF(A345=2,Bed_2_Total+B$347,Bed_2_Total)	
351	Bed_3_Total	=IF(A345=3,Bed_3_Total+B$347,Bed_3_Total)	
352	Bed_4_Total	=IF(A345=4,Bed_4_Total+B$347,Bed_4_Total)	
353	Bed_5_Total	=IF(A345=5,Bed_5_Total+B$347,Bed_5_Total)	
354	Bed_6_Total	=IF(A345=6,Bed_6_Total+B$347,Bed_6_Total)	
355	Bed_7_Total	=IF(A345=7,Bed_7_Total+B$347,Bed_7_Total)	
356	Bed_8_Total	=IF(A345=8,Bed_8_Total+B$347,Bed_8_Total)	
357	Bed_9_Total	=IF(A345=9,Bed_9_Total+B$347,Bed_9_Total)	
358	Bed_10_Total	=IF(A345=10,Bed_10_Total+B$347,Bed_10_Total)	
359			
360		=FORMULA.GOTO(OFFSET(Start_Reference,A345,1),FALSE)	Clear Bed Slot

	A	B	C
1	names	Commands	comments
2			
361		=CLEAR(3)	
362		=SET.VALUE(Beds_Occupied,Beds_Occupied-1)	Free up a Bed
363			
364		=RETURN()	
365			
366	Bed_Routine_B	=IF(Beds_Occupied>=Bed_Number,RETURN())	See If Bed Available
367		=SET.VALUE(Bed_Queue,Bed_Queue-1)	Remove One From Bed Queue
368			
369		=SET.VALUE(Bed_Queue_Counter,Bed_Queue_Counter+1)	
370		=FORMULA(Bed_Queue_Counter,OFFSET(AV2,Bed_Queue_Counter,0))	
371		=FORMULA(Bed_Queue,OFFSET(AV2,Bed_Queue_Counter,1))	
372		=FORMULA(Master_Clock_Time,OFFSET(AV2,Bed_Queue_Counter,2))	
373		=FORMULA(Master_Clock_Time-OFFSET(AV2,Bed_Queue_Counter-1,2),OFFSET(AV2,Bed_Queue_Counter-1,3))	
374		=FORMULA((OFFSET(AV2,Bed_Queue_Counter-1,1)*OFFSET(AV2,Bed_Queue_Counter-1,3)),OFFSET(AV2,Bed_Queue_Cou	
375		=SET.VALUE(Blank_Counter,1)	
376		=WHILE(ISBLANK(OFFSET(Start_Reference,Blank_Counter,1))=FALSE)	Find Next Empty Bed
377		= SET.VALUE(Blank_Counter,Blank_Counter+1)	
378		=NEXT()	
379		=SET.VALUE(B383,End_Simulation_Time*2)	
380		=FOR("True_Counter",1,75)	Find First in Bed Queue
381		= IF(OFFSET(Next_Event_Reference,True_Counter,8)=TRUE)	
382		= OFFSET(Next_Event_Reference,True_Counter,7)	
383		= IF(B382<B383,B382,B383)	Determine its Respective Pointer
384		= END.IF()	
385		=NEXT()	
386		=SET.VALUE(Active_Pointer,MATCH(B383,MCT_State_3,0))	
387	Beds_Occupied	=Beds_Occupied+1	Put Patient in Bed
388		=FORMULA(Master_Clock_Time,OFFSET(Next_Event_Reference,Active_Pointer,8))	Mark Time in Bed
389		=FORMULA(Master_Clock_Time,OFFSET(Start_Reference,Blank_Counter,1))	Mark Time Bed Occupied
390		=FORMULA(4,OFFSET(Next_Event_Reference,Active_Pointer,4))	Make Unit State 4
391		=FORMULA(Blank_Counter,OFFSET(Next_Event_Reference,Active_Pointer,10))	Mark Bed Number
392			
393		=RETURN()	
394			
395			Nurse Routine
396			
397	Nurse_A		Nurse is Complete
398		=FORMULA(OFFSET(Start_Reference,Active_Nurse,2),A399)	Retrieve when Nurse Occupied
399	10072	= Master_Clock_Time-A399	Time Nurse Occupied
400		= FORMULA(3,OFFSET(Next_Event_Reference,Active_Pointer,4))	Make Unit State 3
401		= FORMULA(Master_Clock_Time,OFFSET(Next_Event_Reference,Active_Pointer,7))	Mark Time in Bed Queue
402		= SET.VALUE(Bed_Queue,Bed_Queue+1)	Add One to Bed Queue
403			
404		= IF(Bed_Queue_First=TRUE)	Tabulate Bed Queue Statistics
405		= FORMULA(Bed_Queue_Counter,OFFSET(AV2,Bed_Queue_Counter,0))	
406		= FORMULA(Bed_Queue-1,OFFSET(AV2,Bed_Queue_Counter,1))	
407		= FORMULA(Master_Clock_Time,OFFSET(AV2,Bed_Queue_Counter,2))	
408		= SET.VALUE(Bed_Queue_First,FALSE)	
409		= END.IF()	
410		= SET.VALUE(Bed_Queue_Counter,Bed_Queue_Counter+1)	
411		= FORMULA(Bed_Queue_Counter,OFFSET(AV2,Bed_Queue_Counter,0))	
412		= FORMULA(Bed_Queue,OFFSET(AV2,Bed_Queue_Counter,1))	
413		= FORMULA(Master_Clock_Time,OFFSET(AV2,Bed_Queue_Counter,2))	
414		= FORMULA(Master_Clock_Time-OFFSET(AV2,Bed_Queue_Counter-1,2),OFFSET(AV2,Bed_Queue_Counter-1,3))	
415		= FORMULA((OFFSET(AV2,Bed_Queue_Counter-1,1)*OFFSET(AV2,Bed_Queue_Counter-1,3)),OFFSET(AV2,Bed_Queue	
416			
417		= IF(OFFSET(Next_Event_Reference,Active_Pointer,2)=1)	
418	Nurse_1_Total_1	= IF(Active_Nurse=2,Nurse_1_Total_1+B$399,Nurse_1_Total_1)	Tally Nurse Occupied Time
419	Nurse_2_Total_1	= IF(Active_Nurse=3,Nurse_2_Total_1+B$399,Nurse_2_Total_1)	
420	Nurse_3_Total_1	= IF(Active_Nurse=4,Nurse_3_Total_1+B$399,Nurse_3_Total_1)	
421	Nurse_4_Total_1	= IF(Active_Nurse=5,Nurse_4_Total_1+B$399,Nurse_4_Total_1)	
422		= END.IF()	
423		= IF(OFFSET(Next_Event_Reference,Active_Pointer,2)=2)	
424	Nurse_1_Total_2	= IF(Active_Nurse=2,Nurse_1_Total_2+B$399,Nurse_1_Total_2)	
425	Nurse_2_Total_2	= IF(Active_Nurse=3,Nurse_2_Total_2+B$399,Nurse_2_Total_2)	
426	Nurse_3_Total_2	= IF(Active_Nurse=4,Nurse_3_Total_2+B$399,Nurse_3_Total_2)	
427	Nurse_4_Total_2	= IF(Active_Nurse=5,Nurse_4_Total_2+B$399,Nurse_4_Total_2)	
428		= END.IF()	
429		= IF(OFFSET(Next_Event_Reference,Active_Pointer,2)=3)	
430	Nurse_1_Total_3	= IF(Active_Nurse=2,Nurse_1_Total_3+B$399,Nurse_1_Total_3)	
431	Nurse_2_Total_3	= IF(Active_Nurse=3,Nurse_2_Total_3+B$399,Nurse_2_Total_3)	
432	Nurse_3_Total_3	= IF(Active_Nurse=4,Nurse_3_Total_3+B$399,Nurse_[illegible]otal_3)	
433	Nurse_4_Total_3	= IF(Active_Nurse=5,Nurse_4_Total_3+B$399,Nurse[illegible]Total_3)	
434		= END.IF()	
435		= IF(OFFSET(Next_Event_Reference,Active_Pointer,2)=4)	
436	Nurse_1_Total_4	= IF(Active_Nurse=2,Nurse_1_Total_4+B$399,Nurse_1_Total_4)	
437	Nurse_2_Total_4	= IF(Active_Nurse=3,Nurse_2_Total_4+B$399,Nurse_2_Total_4)	
438	Nurse_3_Total_4	= IF(Active_Nurse=4,Nurse_3_Total_4+B$399,Nurse_3_Total_4)	
439	Nurse_4_Total_4	= IF(Active_Nurse=5,Nurse_4_Total_4+B$399,Nurse_4_Total_4)	
440		= END.IF()	
441		= IF(OFFSET(Next_Event_Reference,Active_Pointer,2)=5)	
442	Nurse_1_Total_5	= IF(Active_Nurse=2,Nurse_1_Total_5+B$399,Nurse_1_Total_5)	
443	Nurse_2_Total_5	= IF(Active_Nurse=3,Nurse_2_Total_5+B$399,Nurse_2_Total_5)	
444	Nurse_3_Total_5	= IF(Active_Nurse=4,Nurse_3_Total_5+B$399,Nurse_3_Total_5)	
445	Nurse_4_Total_5	= IF(Active_Nurse=5,Nurse_4_Total_5+B$399,Nurse_4_Total_5)	
446		= END.IF()	
447		= SET.VALUE(Nurses_Occupied,Nurses_Occupied-1)	Free up Nurse
448			
449		=FORMULA(End_Simulation_Time*2,OFFSET(Unit_Reference,1,Active_Nurse))	Assign Large Time to Nurse
450		=RETURN()	
451			

	A	B	C
1	names	Commands	comments
2			
452	Nurse_B	=IF(Nurses_Occupied>=Nurse_Available,RETURN())	Occupy a Nurse Routine
453	Nurses_Occupied	=Nurses_Occupied+1	Occupy Nurse
454		=SET.VALUE(Registration_Queue,Registration_Queue-1)	Remove One From Registration Qu
455			
456		= SET.VALUE(Reg_Queue_Counter,Reg_Queue_Counter+1)	Tabulate Registration Queue Statis
457		= FORMULA(Reg_Queue_Counter,OFFSET(AP2,Reg_Queue_Counter,0))	
458		= FORMULA(Registration_Queue,OFFSET(AP2,Reg_Queue_Counter,1))	
459		= FORMULA(Master_Clock_Time,OFFSET(AP2,Reg_Queue_Counter,2))	
460		= FORMULA(Master_Clock_Time-OFFSET(AP2,Reg_Queue_Counter-1,2),OFFSET(AP2,Reg_Queue_Counter-1,3))	
461		= FORMULA((OFFSET(AP2,Reg_Queue_Counter-1,1)*OFFSET(AP2,Reg_Queue_Counter-1,3)),OFFSET(AP2,Reg_Queue_	
462		=SET.VALUE(B466,End_Simulation_Time)	
463		=FOR("True_Counter",1,75)	Find First in Registration Queue
464		= IF(OFFSET(Next_Event_Reference,True_Counter,6)=TRUE)	
465		= OFFSET(Next_Event_Reference,True_Counter,5)	
466		= IF(B465<B466,B465,B466)	
467		= END.IF()	
468		=NEXT()	
469		=SET.VALUE(Active_Pointer,MATCH(B466,MCT_State_1,0))	Determine its Respective Pointer
470			
471	Active_Nurse	=MATCH(MAX(Nurse_Box),Nurse_Box,0)+1	Find Nurse Off Longest
472		=FORMULA(Master_Clock_Time,OFFSET(Next_Event_Reference,Active_Pointer,6))	Mark Time in with Nurse
473		=FORMULA(Master_Clock_Time,OFFSET(Start_Reference,Active_Nurse,2))	Mark Time Nurse Occupied
474		=FORMULA(2,OFFSET(Next_Event_Reference,Active_Pointer,4))	Make Unit State 2
475		=SET.VALUE(Patient_Type,OFFSET(Next_Event_Reference,Active_Pointer,2))	Determine Patient Type
476	Time_with_Nurse	=RUN(Nurse_Service)	Determine Nurse Service Time
477		=FORMULA(Time_with_Nurse,OFFSET(Unit_Reference,1,Active_Nurse))	Assign Service Time to Nurse
478		=FORMULA(Active_Pointer,OFFSET(Unit_Reference,2,Active_Nurse))	Assign Pointer to Nurse
479			
480		=RETURN()	
481			
482			Nurse Serve Time Routine
483			
484	Nurse_Service	=RAND()	
485		=IF(Patient_Type=1)	
486	Nurse_Time	= NORMINV(B484,(W124-W118/2)*60,W130)	
487		=END.IF()	
488		=IF(Patient_Type=2)	
489		= SET.VALUE(Nurse_Time,NORMINV(B484,(W125-W119/2)*60,W131))	
490		=END.IF()	
491		=IF(Patient_Type=3)	
492		= SET.VALUE(Nurse_Time,NORMINV(B484,(W126-W120/2)*60,W132))	
493		=END.IF()	
494		=IF(Patient_Type=4)	
495		= SET.VALUE(Nurse_Time,NORMINV(B484,(W127-W121/2)*60,W133))	
496		=END.IF()	
497		=IF(Patient_Type=5)	
498		= SET.VALUE(Nurse_Time,NORMINV(B484,(W128-W122/2)*60,W134))	
499		=END.IF()	
500			
501		=RETURN(Nurse_Time)	
502			
503			Doctor Routine
504			
505	Doctor_A		
506		=FORMULA(OFFSET(Start_Reference,Active_Doctor,3),A507)	Retrieve when Doctor Occupied
507	10048.2679772731	=Master_Clock_Time-A507	Time Doctor Occupied
508			
509		= IF(OFFSET(Next_Event_Reference,Active_Pointer,2)=1)	
510	Doctor_1_Total_1	= IF(Active_Doctor=6,Doctor_1_Total_1+B$399,Doctor_1_Total_1)	Tally Doctor Occupied Time
511	Doctor_2_Total_1	= IF(Active_Doctor=7,Doctor_2_Total_1+B$399,Doctor_2_Total_1)	
512	Doctor_3_Total_1	= IF(Active_Doctor=8,Doctor_3_Total_1+B$399,Doctor_3_Total_1)	
513	Doctor_4_Total_1	= IF(Active_Doctor=9,Doctor_4_Total_1+B$399,Doctor_4_Total_1)	
514		= END.IF()	
515		= IF(OFFSET(Next_Event_Reference,Active_Pointer,2)=2)	
516	Doctor_1_Total_2	= IF(Active_Doctor=6,Doctor_1_Total_2+B$399,Doctor_1_Total_2)	
517	Doctor_2_Total_2	= IF(Active_Doctor=7,Doctor_2_Total_2+B$399,Doctor_2_Total_2)	
518	Doctor_3_Total_2	= IF(Active_Doctor=8,Doctor_3_Total_2+B$399,Doctor_3_Total_2)	
519	Doctor_4_Total_2	= IF(Active_Doctor=9,Doctor_4_Total_2+B$399,Doctor_4_Total_2)	
520		= END.IF()	
521		= IF(OFFSET(Next_Event_Reference,Active_Pointer,2)=3)	
522	Doctor_1_Total_3	= IF(Active_Doctor=6,Doctor_1_Total_3+B$399,Doctor_1_Total_3)	
523	Doctor_2_Total_3	= IF(Active_Doctor=7,Doctor_2_Total_3+B$399,Doctor_2_Total_3)	
524	Doctor_3_Total_3	= IF(Active_Doctor=8,Doctor_3_Total_3+B$399,Doctor_3_Total_3)	
525	Doctor_4_Total_3	= IF(Active_Doctor=9,Doctor_4_Total_3+B$399,Doctor_4_Total_3)	
526		= END.IF()	
527		= IF(OFFSET(Next_Event_Reference,Active_Pointer,2)=4)	
528	Doctor_1_Total_4	= IF(Active_Doctor=6,Doctor_1_Total_4+B$399,Doctor_1_Total_4)	
529	Doctor_2_Total_4	= IF(Active_Doctor=7,Doctor_2_Total_4+B$399,Doctor_2_Total_4	
530	Doctor_3_Total_4	= IF(Active_Doctor=8,Doctor_3_Total_4+B$399,Doctor_3_Total_4)	
531	Doctor_4_Total_4	= IF(Active_Doctor=9,Doctor_4_Total_4+B$399,Doctor_4_Total_4)	
532		= END.IF()	
533		= IF(OFFSET(Next_Event_Reference,Active_Pointer,2)=5)	
534	Doctor_1_Total_5	= IF(Active_Doctor=6,Doctor_1_Total_5+B$399,Doctor_1_Total_5)	
535	Doctor_2_Total_5	= IF(Active_Doctor=7,Doctor_2_Total_5+B$399,Doctor_2_Total_5)	
536	Doctor_3_Total_5	= IF(Active_Doctor=8,Doctor_3_Total_5+B$399,Doctor_3_Total_5)	
537	Doctor_4_Total_5	= IF(Active_Doctor=9,Doctor_4_Total_5+B$399,Doctor_4_Total_5)	
538		= END.IF()	
539			
540		=SET.VALUE(Doctors_Occupied,Doctors_Occupied-1)	Free up Doctor
541		=FORMULA(End_Simulation_Time*2,OFFSET(Unit_Reference,1,Active_Doctor))	Assign Large Time to Doctor
542		=RETURN()	

	A	B	C
1	*names*	Commands	*comments*
2			
543			
544	Doctor_B	=IF(Doctors_Occupied>=Doctor_Available,RETURN())	Occupy a Doctor Routine
545	Doctors_Occupied	=Doctors_Occupied+1	Occupy Doctor
546		=SET.VALUE(B550,End_Simulation_Time)	
547		=FOR("True_Counter",1,75)	Find First in Bed
548		= IF(OFFSET(Next_Event_Reference,True_Counter,9)=TRUE)	
549		= OFFSET(Next_Event_Reference,True_Counter,8)	
550		= IF(B549<B550,B549,B550)	
551		= END.IF()	
552		=NEXT()	
553		=SET.VALUE(Active_Pointer,MATCH(B550,MCT_State_4,0))	Determine its Respective Pointer
554			
555	Active_Doctor	=MATCH(MAX(Doctor_Box),Doctor_Box,0)+5	Find Doctor Off Longest Last Free
556		=FORMULA(Master_Clock_Time,OFFSET(Next_Event_Reference,Active_Pointer,9))	Mark Time in with Doctor
557		=FORMULA(Master_Clock_Time,OFFSET(Start_Reference,Active_Doctor,3))	Mark Time Doctor Occupied
558		=FORMULA(5,OFFSET(Next_Event_Reference,Active_Pointer,4))	Make Unit State 5
559		=SET.VALUE(Patient_Type,OFFSET(Next_Event_Reference,Active_Pointer,2))	Determine Patient Type
560	Time_with_Doctor	=RUN(Doctor_Service)	Determine Doctor Service Time
561		=FORMULA(Time_with_Doctor,OFFSET(Unit_Reference,1,Active_Doctor))	Assign Service Time to Doctor
562		=FORMULA(Active_Pointer,OFFSET(Unit_Reference,2,Active_Doctor))	Assign Pointer to Doctor
563			
564		=RETURN()	
565			
566			Doctor Serve Time Routine
567			
568	Doctor_Service	=RAND()	
569		=IF(Patient_Type=1)	
570	Doctor_Time	= NORMINV(B568,(W171-W165/2)*60,W177)	
571		=END.IF()	
572		=IF(Patient_Type=2)	
573		= SET.VALUE(Doctor_Time,NORMINV(B568,(W172-W166/2)*60,W178))	
574		=END.IF()	
575		=IF(Patient_Type=3)	
576		= SET.VALUE(Doctor_Time,NORMINV(B568,(W173-W167/2)*60,W179))	
577		=END.IF()	
578		=IF(Patient_Type=4)	
579		= SET.VALUE(Doctor_Time,NORMINV(B568,(W172-W166/2)*60,W178))	
580		=END.IF()	
581		=IF(Patient_Type=5)	
582		= SET.VALUE(Doctor_Time,NORMINV(B568,(W172-W166/2)*60,W178))	
583		=END.IF()	
584			
585		=RETURN(Doctor_Time)	
586			
587			Patient Complete Routine
588			
589	Patient_Complete		
590		=SET.VALUE(System_Counter,System_Counter+1)	Tabulate System Statistics
591		=FORMULA(System_Counter,OFFSET(AJ2,System_Counter,0))	
592		=FORMULA(Registration_Queue+Nurses_Occupied+Bed_Queue+Beds_Occupied,OFFSET(AJ2,System_Counter,1))	
593		=FORMULA(Master_Clock_Time,OFFSET(AJ2,System_Counter,2))	
594		=FORMULA(Master_Clock_Time-OFFSET(AJ2,System_Counter-1,2),OFFSET(AJ2,System_Counter-1,3))	
595		=FORMULA((OFFSET(AJ2,System_Counter-1,1)*OFFSET(AJ2,System_Counter-1,3)),OFFSET(AJ2,System_Counter-1,4))	
596			
597		=OFFSET(Next_Event_Reference,Active_Pointer,6)-OFFSET(Next_Event_Reference,Active_Pointer,5)	Time in Queue
598		=OFFSET(Next_Event_Reference,Active_Pointer,7)-OFFSET(Next_Event_Reference,Active_Pointer,6)	Time with Nurse
599		=OFFSET(Next_Event_Reference,Active_Pointer,8)-OFFSET(Next_Event_Reference,Active_Pointer,7)	Time in Bed Queue
600		=Master_Clock_Time-OFFSET(Next_Event_Reference,Active_Pointer,8)	Time in Bed
601		=Master_Clock_Time-OFFSET(Next_Event_Reference,Active_Pointer,9)	Time with Doctor
602		=Master_Clock_Time-OFFSET(Next_Event_Reference,Active_Pointer,3)	Time in System
603			
604		=OFFSET(Next_Event_Reference,Active_Pointer,2)	Get Patient Type
605		=OFFSET(Next_Event_Reference,Active_Pointer,1)	Get Active Unit
606		=FORMULA(B605,OFFSET(Output,B603+1,0))	Active Unit to Output
607		=FORMULA(B604,OFFSET(Output,B603+1,1))	Patient Type to Output
608		=FORMULA(OFFSET(Next_Event_Reference,Active_Pointer,3),OFFSET(Output,B603+1,2))	Enter System to Output
609		=IF(B604=1,0,FORMULA(B597,OFFSET(Output,B603+1,3)))	Registration Queue to Output
610		=IF(B604=1,0,FORMULA(B598,OFFSET(Output,B603+1,4)))	Nurse Output
611		=FORMULA(B599,OFFSET(Output,B603+1,5))	Bed Queue to Output
612		=FORMULA(B600,OFFSET(Output,B603+1,6))	Bed Time to Output
613		=FORMULA(B601,OFFSET(Output,B603+1,7))	Doctor Time to Output
614		=FORMULA(B602,OFFSET(Output,B603+1,8))	System Time to Output
615			
616		=FORMULA.GOTO(OFFSET($E$10,Active_Pointer,1):OFFSET($E$10,Active_Pointer,10),FALSE)	Clear Pointer Next Event Matrix
617		=CLEAR(3)	
618		=VSCROLL(0)	
619		=RETURN()	
620			
621			End Simulation Routine
622			
623	END		
624		=SET.VALUE(B435,0)	Routine to Find Processed Patients
625		=SET.VALUE(B432,0)	
626		=SET.VALUE(B433,0)	
627		=SET.VALUE(Processed_Patients,0)	
628		=FOR("Count",2,Total_Patients)	
629		= IF(ISBLANK(OFFSET(Output,Count,1))=TRUE,NEXT())	
630	Processed_Patients	= Processed_Patients+1	
631		= IF(OFFSET(Output,Count,2)>1)	
632		= IF(OFFSET(Output,Count,3)>0,B632+1,B632)	Total who Wait for Registration
633		= IF(OFFSET(Output,Count,3)=0,B633+1,B633)	Total who Do Not Wait for Reg

	A	B	C
1	names	Commands	comments
2			
634		= END.IF()	
635		= IF(OFFSET(Output,Count,5)>0,B635+1,B635)	Total who Wait for Bed
636		=NEXT()	
637			
638		=FORMULA(Nurse_1_Total_1,OFFSET($BB$1,2,1))	Tally Nurse/Patient Statistics
639		=FORMULA(Nurse_1_Total_2,OFFSET($BB$1,3,1))	
640		=FORMULA(Nurse_1_Total_3,OFFSET($BB$1,4,1))	
641		=FORMULA(Nurse_1_Total_4,OFFSET($BB$1,5,1))	
642		=FORMULA(Nurse_1_Total_5,OFFSET($BB$1,6,1))	
643		=FORMULA(Nurse_2_Total_1,OFFSET($BB$1,2,2))	
644		=FORMULA(Nurse_2_Total_2,OFFSET($BB$1,3,2))	
645		=FORMULA(Nurse_2_Total_3,OFFSET($BB$1,4,2))	
646		=FORMULA(Nurse_2_Total_4,OFFSET($BB$1,5,2))	
647		=FORMULA(Nurse_2_Total_5,OFFSET($BB$1,6,2))	
648		=FORMULA(Nurse_3_Total_1,OFFSET($BB$1,2,3))	
649		=FORMULA(Nurse_3_Total_2,OFFSET($BB$1,3,3))	
650		=FORMULA(Nurse_3_Total_3,OFFSET($BB$1,4,3))	
651		=FORMULA(Nurse_3_Total_4,OFFSET($BB$1,5,3))	
652		=FORMULA(Nurse_3_Total_5,OFFSET($BB$1,6,3))	
653		=FORMULA(Nurse_4_Total_1,OFFSET($BB$1,2,4))	
654		=FORMULA(Nurse_4_Total_2,OFFSET($BB$1,3,4))	
655		=FORMULA(Nurse_4_Total_3,OFFSET($BB$1,4,4))	
656		=FORMULA(Nurse_4_Total_4,OFFSET($BB$1,5,4))	
657		=FORMULA(Nurse_4_Total_5,OFFSET($BB$1,6,4))	
658		=FORMULA(SUM(BC3:BC7),OFFSET($BB$1,7,1))	
659		=FORMULA(SUM(BD3:BD7),OFFSET($BB$1,7,2))	
660		=FORMULA(SUM(BE3:BE7),OFFSET($BB$1,7,3))	
661		=FORMULA(SUM(BF3:BF7),OFFSET($BB$1,7,4))	
662		=Master_Clock_Time	
663		=MOD(Master_Clock_Time,1440)	
664		=IF(W101>W106,W101-2400,W101)	
665		=IF(W102>W107,W102-2400,W102)	
666		=IF(W103>W108,W103-2400,W103)	
667		=IF(W104>W109,W104-2400,W104)	
668		=INT(B$662/1440)*((W106-B664)*0.6)+((B$663-IF(B664>0,B664*0.6,0))-IF(B$663-W106*0.6>0,B$663-W106*0.6,0))	Nurse Work Time
669		=INT(B$662/1440)*((W107-B665)*0.6)+((B$663-IF(B665>0,B665*0.6,0))-IF(B$663-W107*0.6>0,B$663-W107*0.6,0))	
670		=INT(B$662/1440)*((W108-B666)*0.6)+((B$663-IF(B666>0,B666*0.6,0))-IF(B$663-W108*0.6>0,B$663-W108*0.6,0))	
671		=INT(B$662/1440)*((W109-B667)*0.6)+((B$663-IF(B667>0,B667*0.6,0))-IF(B$663-W109*0.6>0,B$663-W109*0.6,0))	
672		=FORMULA(BC$8/B668,BC39)	Nurse Utilization
673		=IF(Nurse_Number>1,FORMULA(BD$8/B669,BC40))	
674		=IF(Nurse_Number>2,FORMULA(BE$8/B670,BC41))	
675		=IF(Nurse_Number>3,FORMULA(BF$8/B671,BC42))	
676			
677		=FORMULA(Doctor_1_Total_1,OFFSET($BB$1,2,5))	Tally Doctor/Patient Statistics
678		=FORMULA(Doctor_1_Total_2,OFFSET($BB$1,3,5))	
679		=FORMULA(Doctor_1_Total_3,OFFSET($BB$1,4,5))	
680		=FORMULA(Doctor_1_Total_4,OFFSET($BB$1,5,5))	
681		=FORMULA(Doctor_1_Total_5,OFFSET($BB$1,6,5))	
682		=FORMULA(Doctor_2_Total_1,OFFSET($BB$1,2,6))	
683		=FORMULA(Doctor_2_Total_2,OFFSET($BB$1,3,6))	
684		=FORMULA(Doctor_2_Total_3,OFFSET($BB$1,4,6))	
685		=FORMULA(Doctor_2_Total_4,OFFSET($BB$1,5,6))	
686		=FORMULA(Doctor_2_Total_5,OFFSET($BB$1,6,6))	
687		=FORMULA(Doctor_3_Total_1,OFFSET($BB$1,2,7))	
688		=FORMULA(Doctor_3_Total_2,OFFSET($BB$1,3,7))	
689		=FORMULA(Doctor_3_Total_3,OFFSET($BB$1,4,7))	
690		=FORMULA(Doctor_3_Total_4,OFFSET($BB$1,5,7))	
691		=FORMULA(Doctor_3_Total_5,OFFSET($BB$1,6,7))	
692		=FORMULA(Doctor_4_Total_1,OFFSET($BB$1,2,8))	
693		=FORMULA(Doctor_4_Total_2,OFFSET($BB$1,3,8))	
694		=FORMULA(Doctor_4_Total_3,OFFSET($BB$1,4,8))	
695		=FORMULA(Doctor_4_Total_4,OFFSET($BB$1,5,8))	
696		=FORMULA(Doctor_4_Total_5,OFFSET($BB$1,6,8))	
697		=FORMULA(SUM(BG3:BG7),OFFSET($BB$1,7,5))	
698		=FORMULA(SUM(BH3:BH7),OFFSET($BB$1,7,6))	
699		=FORMULA(SUM(BI3:BI7),OFFSET($BB$1,7,7))	
700		=FORMULA(SUM(BJ3:BJ7),OFFSET($BB$1,7,8))	
701		=FORMULA(SUM(BC3:BJ3),OFFSET($BB$1,2,9))	Total Doctor & Nurse Statistics
702		=FORMULA(SUM(BC4:BJ4),OFFSET($BB$1,3,9))	
703		=FORMULA(SUM(BC5:BJ5),OFFSET($BB$1,4,9))	
704		=FORMULA(SUM(BC6:BJ6),OFFSET($BB$1,5,9))	
705		=FORMULA(SUM(BC7:BJ7),OFFSET($BB$1,6,9))	
706		=FORMULA(SUM(BK3:BK8),OFFSET($BB$1,7,9))	
707		=IF(W148>W153,W148-2400,W148)	
708		=IF(W149>W154,W149-2400,W149)	
709		=IF(W150>W155,W150-2400,W150)	
710		=IF(W151>W156,W151-2400,W151)	
711		=INT(B$662/1440)*((W153-B707)*0.6)+((B$663-IF(B707>0,B707*0.6,0))-IF(B$663-W153*0.6>0,B$663-W153*0.6,0))	Doctor Work Time
712		=INT(B$662/1440)*((W154-B708)*0.6)+((B$663-IF(B708>0,B708*0.6,0))-IF(B$663-W154*0.6>0,B$663-W154*0.6,0))	
713		=INT(B$662/1440)*((W155-B709)*0.6)+((B$663-IF(B709>0,B709*0.6,0))-IF(B$663-W155*0.6>0,B$663-W155*0.6,0))	
714		=INT(B$662/1440)*((W156-B710)*0.6)+((B$663-IF(B710>0,B710*0.6,0))-IF(B$663-W156*0.6>0,B$663-W156*0.6,0))	
715		=FORMULA(BG$8/B711,BC43)	Doctor Utilization
716		=IF(Doctor_Number>1,FORMULA(BH$8/B712,BC44))	
717		=IF(Doctor_Number>2,FORMULA(BI$8/B713,BC45))	
718		=IF(Doctor_Number>3,FORMULA(BJ$8/B714,BC46))	
719			
720		=FORMULA(Bed_1_Total,OFFSET($BB$1,10,1))	Tally Bed Statistics
721		=FORMULA(Bed_2_Total,OFFSET($BB$1,11,1))	
722		=FORMULA(Bed_3_Total,OFFSET($BB$1,12,1))	
723		=FORMULA(Bed_4_Total,OFFSET($BB$1,13,1))	
724		=FORMULA(Bed_5_Total,OFFSET($BB$1,14,1))	

	A	B	C
1	Names	Commands	comments
2			
725		=FORMULA(Bed_6_Total,OFFSET($BB$1,15,1))	
726		=FORMULA(Bed_7_Total,OFFSET($BB$1,16,1))	
727		=FORMULA(Bed_8_Total,OFFSET($BB$1,17,1))	
728		=FORMULA(Bed_9_Total,OFFSET($BB$1,18,1))	
729		=FORMULA(Bed_10_Total,OFFSET($BB$1,19,1))	
730		=FORMULA(BC11/Master_Clock_Time,BD11)	Bed Utilization
731		=IF(Bed_Number>1,FORMULA(BC12/Master_Clock_Time,BD12))	
732		=IF(Bed_Number>2,FORMULA(BC13/Master_Clock_Time,BD13))	
733		=IF(Bed_Number>3,FORMULA(BC14/Master_Clock_Time,BD14))	
734		=IF(Bed_Number>4,FORMULA(BC15/Master_Clock_Time,BD15))	
735		=IF(Bed_Number>5,FORMULA(BC16/Master_Clock_Time,BD16))	
736		=IF(Bed_Number>6,FORMULA(BC17/Master_Clock_Time,BD17))	
737		=IF(Bed_Number>7,FORMULA(BC18/Master_Clock_Time,BD18))	
738		=IF(Bed_Number>8,FORMULA(BC19/Master_Clock_Time,BD19))	
739		=IF(Bed_Number>9,FORMULA(BC20/Master_Clock_Time,BD20))	
740		=FORMULA(SUM(AZ3:OFFSET(AV3,Bed_Queue_Counter,4))/(Master_Clock_Time-AX3),BC28)	Average Bed Queue Length
741		=FORMULA(SUM(AZ3:OFFSET(AV3,Bed_Queue_Counter,4))/Processed_Patients,BC29)	Average Time in Bed Queue
742		=FORMULA(MAX(AE3:OFFSET(AE3,Processed_Patients,0)),BC30)	Maximum Time in Bed Queue
743		=FORMULA(MAX(AF3:OFFSET(AF3,Processed_Patients,0)),BC31)	Maximum Time in Bed
744		=FORMULA(B635/Processed_Patients,BC32)	Fraction who Wait for Bed
745			
746		=FORMULA(SUM(AT3:OFFSET(AP3,Reg_Queue_Counter,4))/(Master_Clock_Time-AR3),BC24)	Average Registration Queue Length
747		=FORMULA(SUM(AT3:OFFSET(AP3,Reg_Queue_Counter,4))/(B632+B633),BC25)	Average Time in Registration Queue
748		=FORMULA(MAX(AC3:OFFSET(AC3,Processed_Patients,0)),BC26)	Maximum Time in Queue
749		=FORMULA(B632/(B632+B633),BC27)	Fraction who Wait for Registration
750			
751		=FORMULA(SUM(AN3:OFFSET(AJ3,System_Counter,4))/(Master_Clock_Time-AL3),BC33)	Average Patients in System
752		=FORMULA(MAX(AH3:OFFSET(AH3,Processed_Patients,0)),BC34)	Maximum Time in System
753			
754		=HISTOGRAM(AC4:OFFSET(AC4,Processed_Patients,0),BG13,BF14:BF103)	Frequency Distributions
755		=HISTOGRAM(AD4:OFFSET(AD4,Processed_Patients,0),BI13,BF14:BF103)	
756		=HISTOGRAM(AE4:OFFSET(AE4,Processed_Patients,0),BK13,BF14:BF103)	
757		=HISTOGRAM(AF4:OFFSET(AF4,Processed_Patients,0),BM13,BF14:BF103)	
758		=HISTOGRAM(AG4:OFFSET(AG4,Processed_Patients,0),BO13,BF14:BF103)	
759		=HISTOGRAM(AH4:OFFSET(AH4,Processed_Patients,0),BQ13,BF14:BF103)	
760			
761		=COPY(AQ4:OFFSET(AQ4,Reg_Queue_Counter,0),'D:\THESIS\EXCELFILES\QUEUING\Q-MASTER.XLS'!$AT$12)	Update Information in Q-Master
762		=COPY(AR4:OFFSET(AR4,Reg_Queue_Counter,0),'D:\THESIS\EXCELFILES\QUEUING\Q-MASTER.XLS'!$AU$12)	
763		=COPY(AW4:OFFSET(AW4,Bed_Queue_Counter,0),'D:\THESIS\EXCELFILES\QUEUING\Q-MASTER.XLS'!$AW$12)	
764		=COPY(AX4:OFFSET(AX4,Bed_Queue_Counter,0),'D:\THESIS\EXCELFILES\QUEUING\Q-MASTER.XLS'!$AX$12)	
765		=COPY(AK4:OFFSET(AK4,System_Counter,0),'D:\THESIS\EXCELFILES\QUEUING\Q-MASTER.XLS'!$AZ$12)	
766		=COPY(AL4:OFFSET(AL4,System_Counter,0),'D:\THESIS\EXCELFILES\QUEUING\Q-MASTER.XLS'!$BA$12)	
767			
768		=ACTIVATE("Q-Master.XLS")	
769			
770		=DEFINE.NAME("Reg_Queue_Number",'D:\THESIS\EXCELFILES\QUEUING\Q-MASTER.XLS'!$AT$11:OFFSET('D:\THESIS\	
771		=DEFINE.NAME("Reg_Queue_Start",'D:\THESIS\EXCELFILES\QUEUING\Q-MASTER.XLS'!$AU$11:OFFSET('D:\THESIS\EX	
772		=DEFINE.NAME("Bed_Queue_Number",'D:\THESIS\EXCELFILES\QUEUING\Q-MASTER.XLS'!$AW$11:OFFSET('D:\THESIS	
773		=DEFINE.NAME("Bed_Queue_Start",'D:\THESIS\EXCELFILES\QUEUING\Q-MASTER.XLS'!$AX$11:OFFSET('D:\THESIS\EX	
774		=DEFINE.NAME("System_Number",'D:\THESIS\EXCELFILES\QUEUING\Q-MASTER.XLS'!$AZ$11:OFFSET('D:\THESIS\EXC	
775		=DEFINE.NAME("System_Start",'D:\THESIS\EXCELFILES\QUEUING\Q-MASTER.XLS'!$BA$11:OFFSET('D:\THESIS\EXCEL	
776		=CHANGE.LINK("Queuing.XLM","Queuing.XLM",1)	
777			
778		=RETURN()	

	Q	R	S	T	U	V	W	X
1	DIALOGS							
2	type	x	y	wide	high	text	init/result	*names*
4	INPUT_BOX_1	200	98	600	338	Initial Simulation Input Information		
5	1	460	16	88		Done		
6	5	53	17			Number of Iterations (Hours)		
7	7	295	15	60			10080	End_Simulation_Tim
8	5	150	31			or		
9	5	50	52			Total Patients through System		
10	7	295	48	60			999	Maximum_Units
11	5	50	84			Number of Beds (Maximum 10)		
12	7	295	80	60			5	Bed_Number
13	14	47	130	515	187	Mean Time Between Arrival (Poisson)		
14	5	162	146			Times are Base Upon a 24 Hour Clock		
15	5	55	169			0000		
16	8	112	167	60		0	0.3	
17	5	55	194			0100		
18	8	112	192	60		1	0.35	
19	5	55	219			0200		
20	8	112	217	60		2	0.4	
21	5	55	244			0300		
22	8	112	242	60		3	0.45	
23	5	55	269			0400		
24	8	112	267	60		4	0.5	
25	5	55	294			0500		
26	8	112	292	60		5	0.45	
27	5	187	169			0600		
28	8	242	167	60		6	0.4	
29	5	187	194			0700		
30	8	242	192	60		7	0.35	
31	5	187	219			0800		
32	8	242	217	60		8	0.3	
33	5	187	244			0900		
34	8	242	242	60		9	0.275	
35	5	187	269			1000		
36	8	242	267	60		10	0.25	
37	5	187	294			1100		
38	8	242	292	60		11	0.225	
39	5	312	169			1200		
40	8	362	167	60		12	0.2	
41	5	312	194			1300		
42	8	362	192	60		13	0.225	
43	5	312	219			1400		
44	8	362	217	60		14	0.25	
45	5	312	244			1500		
46	8	362	242	60		15	0.275	
47	5	312	269			1600		
48	8	362	267	60		16	0.3	
49	5	312	294			1700		
50	8	362	292	60		17	0.3	
51	5	432	169			1800		
52	8	487	167	60		18	0.3	
53	5	432	194			1900		
54	8	487	192	60		19	0.3	
55	5	432	219			2000		
56	8	487	217	60		20	0.3	
57	5	432	244			2100		
58	8	487	242	60		21	0.3	
59	5	432	269			2200		
60	8	487	267	60		22	0.3	
61	5	432	294			2300		
62	8	487	292	60		23	0.3	
63	2	461	44	88		Cancel		
64						24	=W16	

	Q	R	S	T	U	V	W	X
1	DIALOGS							
2	type	x	y	wide	high	text	init/result	*names*
66	INPUT_BOX_2	196	115	616	259	Patient Input Information		
67	14	36	13	400	220	Types of Patients		
68	1	491	19	88		Done		
69	5	56	53			Category		
70	5	86	83			1		
71	5	86	113			2		
72	5	86	143			3		
73	5	86	173			4		
74	5	86	203			5		
75	5	161	53			Description		
76	5	161	83			Open Wounds		
77	5	161	113			Closed Injuries		
78	5	161	143			Multiple Tram		
79	5	161	173			Visceral Complaints		
80	5	161	203			Chronic Complaints		
81	5	336	53			Frequency		
82	7	351	83	50			8	Patient_Freq_1
83	7	351	113	50			13	Patient_Freq_2
84	7	351	143	50			33	Patient_Freq_3
85	7	351	173	50			20	Patient_Freq_4
86	7	351	203	50			26	Patient_Freq_5
87	2	493	47	88		Cancel		
88								
89								
90	INPUT_BOX_3	168	116	840	280	Nurse Input Information		
91	1	705	11	88		Done		
92	14	471	93	329	158	Nurse Information		
93	5	483	129			Total Nurses		
94	11						2	Nurse_Number
95	12	517	146		23	1		
96	12					2		
97	12					3		
98	12					4		
99	5	643	109			Shift Times		
100	5	640	130			On		
101	8	630	146	50			0	Nurse_1_On
102	8	630	171	50			800	Nurse_2_On
103	8	630	194	50			0	Nurse_3_On
104	8	630	217	50			0	Nurse_4_On
105	5	715	130			Off		
106	8	705	146	50			2400	Nurse_1_Off
107	8	705	171	50			1600	Nurse_2_Off
108	8	705	194	50			0	Nurse_3_Off
109	8	705	217	50			0	Nurse_4_Off
110	14	33	32	413	220	Time with Patient (Normally Distributed)		
111	5	55	72			Category		
112	5	85	104			1		
113	5	85	134			2		
114	5	85	164			3		
115	5	85	194			4		
116	5	85	224			5		
117	5	149	72			Lower		
118	8	148	102	50			0	
119	8	148	132	50			0.2	
120	8	148	162	50			0.15	
121	8	148	192	50			0.1	
122	8	148	222	50			0.05	
123	5	249	72			Upper		
124	8	248	102	50			0	
125	8	248	132	50			0.3	
126	8	248	162	50			0.25	
127	8	248	192	50			0.2	
128	8	248	222	50			0.15	
129	5	336	72			Std. Dev.		
130	8	346	102	50			1	
131	8	346	132	50			1	
132	8	346	162	50			1	
133	8	346	192	50			1	
134	8	346	222	50			1	
135	2	706	44	88		Cancel		

	Q	R	S	T	U	V	W	X
1	DIALOGS							
2	type	x	y	wide	high	text	init/result	*names*
137	INPUT_BOX_1	168	116	840	280	Doctor Input Information		
138	1	705	11	88		Done		
139	14	471	93	329	158	Doctor Information		
140	5	483	129			Total Doctors		
141	11						2	Doctor_Number
142	12	517	146		23	1		
143	12					2		
144	12					3		
145	12					4		
146	5	643	109			Shift Times		
147	5	640	130			On		
148	8	630	146	50			0	Doctor_1_On
149	8	630	171	50			800	Doctor_2_On
150	8	630	194	50			0	Doctor_3_On
151	8	630	217	50			0	Doctor_4_On
152	5	715	130			Off		
153	8	705	146	50			2400	Doctor_1_Off
154	8	705	171	50			1600	Doctor_2_Off
155	8	705	194	50			0	Doctor_3_Off
156	8	705	217	50			0	Doctor_4_Off
157	14	33	32	413	220	Time with Patient (Normally Distributed)		
158	5	55	72			Category		
159	5	85	104			1		
160	5	85	134			2		
161	5	85	164			3		
162	5	85	194			4		
163	5	85	224			5		
164	5	149	72			Lower		
165	8	148	102	50			0.25	
166	8	148	132	50			0.2	
167	8	148	162	50			0.15	
168	8	148	192	50			0.1	
169	8	148	222	50			0.05	
170	5	249	72			Upper		
171	8	248	102	50			0.35	
172	8	248	132	50			0.3	
173	8	248	162	50			0.25	
174	8	248	192	50			0.2	
175	8	248	222	50			0.15	
176	5	336	72			Std. Dev.		
177	8	346	102	50			1	
178	8	346	132	50			1	
179	8	346	162	50			1	
180	8	346	192	50			1	
181	8	346	222	50			1	
182	2	706	44	88		Cancel		

APPENDIX C
CHAMPUS INPUT & MONTE CARLO MACRO

Var. Ref.	type	location	high / opt / mean / 1	low / pes / SD / 2	most / 3	4	5	6	7	8	9
data	2	1.286	0.5	1.3	2.5						
name / ref		R8C20									
number	1										
data	2	1.393	0.5	1.3	2.5						
name / ref		R9C20									
number	2										
data	2	1.421	0.5	1.3	2.5						
name / ref		R10C20									
number	3										
data	2	2.047	1	2.3	4						
name / ref		R15C20									
number	4										
data	2	2.354	1	2.3	4						
name / ref		R16C20									
number	5										
data	2	2.436	1	2.3	4						
name / ref		R17C20									
number	6										
data	2	4.109	2	4	6						
name / ref		R22C20									
number	7										
data	2	3.411	2	4	6						
name / ref		R23C20									
number	8										
data	2	3.917	2	4	6						
name / ref		R24C20									
number	9										
data	4	0.15	0.13	0.135	0.14	0.145	0.15	0.155	0.16	0.165	0.17
name / ref		R30C22	20	15	10	10	10	10	10	5	5
number	10	95	0.210526316	0.368421053	0.473684	0.578947	0.684211	0.789474	0.894737	0.947368	1
data	4	0.15	0.13	0.135	0.14	0.145	0.15	0.155	0.16	0.165	0.17
name / ref		R31C22	20	15	10	10	10	10	10	5	5
number	11	95	0.210526316	0.368421053	0.473684	0.578947	0.684211	0.789474	0.894737	0.947368	1
data	4	0.15	0.13	0.135	0.14	0.145	0.15	0.155	0.16	0.165	0.17
name / ref		R32C22	20	15	10	10	10	10	10	5	5
number	12	95	0.210526316	0.368421053	0.473684	0.578947	0.684211	0.789474	0.894737	0.947368	1
data	4	0.15	0.13	0.135	0.14	0.145	0.15	0.155	0.16	0.165	0.17
name / ref		R36C22	20	15	10	10	10	10	10	5	5
number	13	95	0.210526316	0.368421053	0.473684	0.578947	0.684211	0.789474	0.894737	0.947368	1
data	4	0.15	0.13	0.135	0.14	0.145	0.15	0.155	0.16	0.165	0.17
name / ref		R37C22	20	15	10	10	10	10	10	5	5
number	14	95	0.210526316	0.368421053	0.473684	0.578947	0.684211	0.789474	0.894737	0.947368	1
data	4	0.15	0.13	0.135	0.14	0.145	0.15	0.155	0.16	0.165	0.17
name / ref		R38C22	20	15	10	10	10	10	10	5	5
number	15	95	0.210526316	0.368421053	0.473684	0.578947	0.684211	0.789474	0.894737	0.947368	1
data	4	0.15	0.13	0.135	0.14	0.145	0.15	0.155	0.16	0.165	0.17
name / ref		R43C22	20	15	10	10	10	10	10	5	5
number	16	95	0.210526316	0.368421053	0.473684	0.578947	0.684211	0.789474	0.894737	0.947368	1
data	4	0.15	0.13	0.135	0.14	0.145	0.15	0.155	0.16	0.165	0.17
name / ref		R44C22	20	15	10	10	10	10	10	5	5
number	17	95	0.210526316	0.368421053	0.473684	0.578947	0.684211	0.789474	0.894737	0.947368	1
data	4	0.15	0.13	0.135	0.14	0.145	0.15	0.155	0.16	0.165	0.17
name / ref		R45C22	20	15	10	10	10	10	10	5	5
number	18	95	0.210526316	0.368421053	0.473684	0.578947	0.684211	0.789474	0.894737	0.947368	1

data	2	0	0.03	0.07	0.15						
name / ref		R30C24									
number	19										
data	2	0	0.03	0.07	0.15						
name / ref		R31C24									
number	20										
data	2	0	0.03	0.07	0.15						
name / ref		R32C24									
number	21										
data	2	0	0.03	0.07	0.15						
name / ref		R36C24									
number	22										
data	2	0	0.03	0.07	0.15						
name / ref		R37C24									
number	23										
data	2	0	0.03	0.07	0.15						
name / ref		R38C24									
number	24										
data	2	0	0.03	0.07	0.15						
name / ref		R43C24									
number	25										
data	2	0	0.03	0.07	0.15						
name / ref		R44C24									
number	26										
data	2	0	0.03	0.07	0.15						
name / ref		R45C24									
number	27										
data	1	902969	925000	875000							
name / ref		R8C27									
number	28										
data	1	824250	850000	800000							
name / ref		R9C27									
number	29										
data	1	311754	325000	300000							
name / ref		R10C27									
number	30										
data	1	902969	925000	875000							
name / ref		R15C27									
number	31										
data	1	824250	850000	800000							
name / ref		R16C27									
number	32										
data	1	311754	325000	300000							
name / ref		R17C27									
number	33										
data	1	902969	925000	875000							
name / ref		R22C27									
number	34										
data	1	824250	850000	775000							
name / ref		R23C27									
number	35										
data	1	311754	325000	300000							
name / ref		R24C27									
number	36										
data	1	908609	925000	875000							
name / ref		R8C28									
number	37										
data	1	896762	925000	875000							
name / ref		R8C31									
number	38										
data	1	870088	900000	825000							
name / ref		R9C28									
number	39										
data	1	909335	925000	875000							
name / ref		R9C31									
number	40										
data	1	329277	350000	300000							
name / ref		R10C28									
number	41										

data	1	344147	370000	330000							
name / ref		R10C31									
number	42										
data	1	908609	920000	880000							
name / ref		R15C28									
number	43										
data	1	896762	910000	870000							
name / ref		R15C31									
number	44										
data	1	870088	900000	850000							
name / ref		R16C28									
number	45										
data	1	909335	920000	875000							
name / ref		R16C31									
number	46										
data	1	329277	335000	300000							
name / ref		R17C28									
number	47										
data	1	344147	375000	325000							
name / ref		R17C31									
number	48										
data	1	908609	930000	880000							
name / ref		R22C28									
number	49										
data	1	896762	905000	870000							
name / ref		R22C31									
number	50										
data	1	870088	900000	840000							
name / ref		R23C28									
number	51										
data	1	909335	930000	880000							
name / ref		R23C31									
number	52										
data	1	329277	350000	300000							
name / ref		R24C28									
number	53										
data	1	344147	375000	320000							
name / ref		R24C31									
number	54										

MONTE CARLO MACRO

	A	B	C
1	names	Commands	comments
2			
3		Summary Information	
4	Title:	Monte Carlo Macro	
5	Version:	v1.0	
6	Author	Dixon Hicks	
7	Corporation:	Naval Postgraduate School	
8	Creation Date:	Tuesday March 2, 1993	
9			
10	CommandWindow	CommandWindow	
11	Monte_Carlo		
12		=SET.VALUE(First_Iteration,FALSE)	
13		=SET.VALUE(Variables_Known,TRUE)	
14		=SET.VALUE(Recall_Input,FALSE)	
15		=SET.VALUE(Screen_Update,TRUE)	
16		=SET.VALUE(Total_Objective,0)	
17		=SET.VALUE(Total_Squared,0)	
18		=DIALOG.BOX(Input_One)	Enter Initial Data
19		=IF(B18=FALSE)	
20	QUIT	= ALERT("Simulation Cancelled",3)	End Simulation by user
21		= MESSAGE(FALSE)	
22		= HALT()	
23		=END.IF()	
24		=IF(Recall_Input=TRUE)	Copy Old Input Data
25		= COPY(ABSREF(RELREF(OFFSET(ABSREF(K11,!$A$1),1,0),!$A$1),!$A$1):OFFSET(ABSREF(K11,!$A$1),180,11),	if Available
26		= COPY(ABSREF(RELREF(OFFSET(ABSREF(K11,!$A$1),0,13),!$A$1),!$A$1):OFFSET(ABSREF(K11,!$A$1),64,13),	
27		= COPY(ABSREF(RELREF(OFFSET(ABSREF(K11,!$A$1),0,12),!$A$1),!$A$1):OFFSET(ABSREF(K11,!$A$1),26,12),	
28		= GOTO(B18)	
29		=END.IF()	
30			
31		=IF(First_Iteration=TRUE)	Clear Variables if First Run
32		= FORMULA.GOTO(Variable_Box,TRUE)	
33		= CLEAR(3)	
34		=END.IF()	
35		=FORMULA.GOTO(Output_Box,TRUE)	
36		=CLEAR(3)	
37		=VSCROLL(1,TRUE)	
38			
39	Divisions	=(Upper_Limit-Lower_Limit)/Number_to_Display	Determine Freq Distribution
40		=FOR("Count",0,Number_to_Display)	
41		= FORMULA(Lower_Limit+(Count*B39),OFFSET(Output_Ref,Count+2,1))	
42		=NEXT()	
43		=ACTIVATE.PREV()	
44		=IF(Variables_Known,GOTO(Iterations))	
45			
46			
47			
48		=SET.VALUE(Variable_Count,0)	Enter Variable Information
49		=FOR("Count",1,Number_Variables)	
50	Variable_Count	=Variable_Count+1	
51		=IF(OFFSET(Var_Ref,(Variable_Count*3)-1,1)="",SET.VALUE(K42,#N/A))	
52		=IF(OFFSET(Var_Ref,(Variable_Count*3)-1,2)="",SET.VALUE(K39,""))	
53		=FORMULA(Variable_Count,K98)	
54		=DIALOG.BOX(Input_Two)	Initial Input
55		=IF(Distribution_Type=1)	Activate Appropriate
56		= FORMULA($E$50-200,$E$50)	Distribution Box
57		= FORMULA($E$51-200,$E$51)	
58		= FORMULA($E$53-200,$E$53)	
59		=ELSE.IF(Distribution_Type=2)	
60		= FORMULA($E$55-200,$E$55)	
61		= FORMULA($E$56-200,$E$56)	
62		= FORMULA($E$58-200,$E$58)	
63		= FORMULA($E$60-200,$E$60)	
64		=END.IF()	
65		=IF(Distribution_Type=3)	
66		= FORMULA($E$62-200,$E$62)	
67		= FORMULA($E$63-200,$E$63)	
68		= FORMULA($E$65-200,$E$65)	
69		=ELSE.IF(Distribution_Type=4)	
70		= FORMULA($E67-200,$E67)	
71		= FORMULA($E68-200,$E68)	
72		= FORMULA($E69-200,$E69)	
73		= FORMULA($E70-200,$E70)	
74		= FORMULA($E71-200,$E71)	
75		= FORMULA($E72-200,$E72)	
76		= FORMULA($E73-200,$E73)	
77		= FORMULA($E74-200,$E74)	
78		= FORMULA($E75-200,$E75)	

	A	B	C
1	names	Commands	comments
2			
79		= FORMULA($E76-200,$E76)	
80		= FORMULA($E77-200,$E77)	
81		= FORMULA($E78-200,$E78)	
82		=END.IF()	
83		=DIALOG.BOX(Input_Two)	Distribution Input
84		=FORMULA(214,$E$50)	Deactivate Distribution Boxes
85		=FORMULA(205,$E$51)	
86		=FORMULA(205,$E$53)	
87		=FORMULA(214,$E$55)	
88		=FORMULA(205,$E$56)	
89		=FORMULA(205,$E$58)	
90		=FORMULA(205,$E$60)	
91		=FORMULA(214,$E$62)	
92		=FORMULA(205,$E$63)	
93		=FORMULA(205,$E$65)	
94		=FORMULA(214,$E67)	
95		=FORMULA(205,$E68)	
96		=FORMULA(205,$E69)	
97		=FORMULA(205,$E70)	
98		=FORMULA(205,$E71)	
99		=FORMULA(205,$E72)	
100		=FORMULA(205,$E73)	
101		=FORMULA(205,$E74)	
102		=FORMULA(205,$E75)	
103		=FORMULA(205,$E76)	
104		=FORMULA(205,$E77)	
105		=FORMULA(205,$E78)	
106			
107		=IF(K42>0,SET.VALUE(Variable_Reference,""))	If Variable Name Given
108			Clear Reference
109			
110		=FORMULA(Variable_Count,OFFSET(Var_Ref,(Variable_Count*3),1))	Record Variable Number
111		=FORMULA(Distribution_Type,OFFSET(Var_Ref,(Variable_Count*3-2),1))	Record Distribution Type
112			
113		=IF(Variable_Reference="")	Determine Link to Worksheet
114		= FORMULA(INDEX(NAMES(),,Variable_Name),OFFSET(Var_Ref,(Variable_Count*3-1),1))	
115		= FORMULA.GOTO(GET.CELL(5,OFFSET(Var_Ref,(Variable_Count*3-1),1)))	
116		= REFTEXT(ACTIVE.CELL(),FALSE)	
117		= FORMULA(B116,OFFSET(Var_Ref,(Variable_Count*3-1),2))	
118		= REPLACE(OFFSET(Var_Ref,(Variable_Count*3-1),2),1,FIND("!",OFFSET(Var_Ref,(Variable_Count*3-1),2)),"")	
119		= FORMULA(B118,OFFSET(Var_Ref,(Variable_Count*3-1),2))	
120		=ELSE()	
121		= FORMULA(Variable_Reference,OFFSET(Var_Ref,(Variable_Count*3-1),2))	
122		=END.IF()	
123			
124		=ABSREF(OFFSET(Var_Ref,(Variable_Count*3-1),2),!$A$1)	Obtain Original Variable Value
125		=FORMULA(B124,OFFSET(Var_Ref,(Variable_Count*3-2),2))	Record Variable Value
126			
127		=IF(Distribution_Type=1)	Record Distribution Statistics
128		= FORMULA(K54,OFFSET(Var_Ref,(Variable_Count*3)-2,3))	to Variable Box
129		= FORMULA(K52,OFFSET(Var_Ref,(Variable_Count*3)-2,4))	
130		=ELSE.IF(Distribution_Type=2)	
131		= FORMULA(K61,OFFSET(Var_Ref,(Variable_Count*3)-2,3))	
132		= FORMULA(K59,OFFSET(Var_Ref,(Variable_Count*3)-2,4))	
133		= FORMULA(K57,OFFSET(Var_Ref,(Variable_Count*3)-2,5))	
134		=END.IF()	
135		=IF(Distribution_Type=3)	
136		= FORMULA(K64,OFFSET(Var_Ref,(Variable_Count*3)-2,3))	
137		= FORMULA(K66,OFFSET(Var_Ref,(Variable_Count*3)-2,4))	
138		=ELSE.IF(Distribution_Type=4)	
139		= FORMULA(K79,OFFSET(Var_Ref,(Variable_Count*3)-2,3))	
140		= FORMULA(K80,OFFSET(Var_Ref,(Variable_Count*3)-1,3))	
141		= FORMULA(K81,OFFSET(Var_Ref,(Variable_Count*3)-2,4))	
142		= FORMULA(K82,OFFSET(Var_Ref,(Variable_Count*3)-1,4))	
143		= FORMULA(K83,OFFSET(Var_Ref,(Variable_Count*3)-2,5))	
144		= FORMULA(K84,OFFSET(Var_Ref,(Variable_Count*3)-1,5))	
145		= FORMULA(K85,OFFSET(Var_Ref,(Variable_Count*3)-2,6))	
146		= FORMULA(K86,OFFSET(Var_Ref,(Variable_Count*3)-1,6))	
147		= FORMULA(K87,OFFSET(Var_Ref,(Variable_Count*3)-2,7))	
148		= FORMULA(K88,OFFSET(Var_Ref,(Variable_Count*3)-1,7))	
149		= FORMULA(K89,OFFSET(Var_Ref,(Variable_Count*3)-2,8))	
150		= FORMULA(K90,OFFSET(Var_Ref,(Variable_Count*3)-1,8))	
151		= FORMULA(K91,OFFSET(Var_Ref,(Variable_Count*3)-2,9))	
152		= FORMULA(K92,OFFSET(Var_Ref,(Variable_Count*3)-1,9))	
153		= FORMULA(K93,OFFSET(Var_Ref,(Variable_Count*3)-2,10))	
154		= FORMULA(K94,OFFSET(Var_Ref,(Variable_Count*3)-1,10))	
155		= FORMULA(K95,OFFSET(Var_Ref,(Variable_Count*3)-2,11))	
156		= FORMULA(K96,OFFSET(Var_Ref,(Variable_Count*3)-1,11))	
157		= SUM(OFFSET(Var_Ref,(Variable_Count*3)-1,3) OFFSET(Var_Ref,(Variable_Count*3)-1,11))	
158		= FORMULA(B157,OFFSET(Var_Ref,(Variable_Count*3),2))	

	A	B	C
1	names	Commands	comments
2			
159		= FORMULA(OFFSET(Var_Ref,(Variable_Count*3)-1,3)/B$157,OFFSET(Var_Ref,(Variable_Count*3),3))	
160		= OFFSET(Var_Ref,(Variable_Count*3)-1,4)/B$157+OFFSET(Var_Ref,(Variable_Count*3),3)	
161		= FORMULA(B160,OFFSET(Var_Ref,(Variable_Count*3),4))	
162		= OFFSET(Var_Ref,(Variable_Count*3)-1,5)/B$157+OFFSET(Var_Ref,(Variable_Count*3),4)	
163		= FORMULA(B162,OFFSET(Var_Ref,(Variable_Count*3),5))	
164		= OFFSET(Var_Ref,(Variable_Count*3)-1,6)/B$157+OFFSET(Var_Ref,(Variable_Count*3),5)	
165		= FORMULA(B164,OFFSET(Var_Ref,(Variable_Count*3),6))	
166		= OFFSET(Var_Ref,(Variable_Count*3)-1,7)/B$157+OFFSET(Var_Ref,(Variable_Count*3),6)	
167		= FORMULA(B166,OFFSET(Var_Ref,(Variable_Count*3),7))	
168		= OFFSET(Var_Ref,(Variable_Count*3)-1,8)/B$157+OFFSET(Var_Ref,(Variable_Count*3),7)	
169		= FORMULA(B168,OFFSET(Var_Ref,(Variable_Count*3),8))	
170		= OFFSET(Var_Ref,(Variable_Count*3)-1,9)/B$157+OFFSET(Var_Ref,(Variable_Count*3),8)	
171		= FORMULA(B170,OFFSET(Var_Ref,(Variable_Count*3),9))	
172		= OFFSET(Var_Ref,(Variable_Count*3)-1,10)/B$157+OFFSET(Var_Ref,(Variable_Count*3),9)	
173		= FORMULA(B172,OFFSET(Var_Ref,(Variable_Count*3),10))	
174		= OFFSET(Var_Ref,(Variable_Count*3)-1,11)/B$157+OFFSET(Var_Ref,(Variable_Count*3),10)	
175		= FORMULA(B174,OFFSET(Var_Ref,(Variable_Count*3),11))	
176		=END.IF()	
177		=NEXT()	
178			
179			
180			
181	Iterations	=ECHO(Screen_Update)	
182		=FOR("Iteration_Count",1,Number_Iterations)	Do Monte Carlo Simulation
183		= MESSAGE(TRUE,Iteration_Count)	Display Iteration Number
184		= SET.VALUE(Variable_Count,0)	
185		= FOR("Count",1,Number_Variables)	Call up Each Variable and
186		= SET.VALUE(Variable_Count,Variable_Count+1)	Calculate a Probabilities
187		= SET.VALUE(Distribution_Type,OFFSET(Var_Ref,(Variable_Count*3-2),1))	Value, then Record to
188		= IF(Distribution_Type=1)	Worksheet
189		= RAND()	
190		= OFFSET(Var_Ref,(Variable_Count*3)-2,3)	
191		= OFFSET(Var_Ref,(Variable_Count*3)-2,4)	
192		= (B190-B191)*B189+B191	
193		= FORMULA(B192,ABSREF(OFFSET(Var_Ref,(Variable_Count*3-1),2),!$A$1))	
194		= ELSE.IF(Distribution_Type=2)	
195		= RAND()	
196		= OFFSET(Var_Ref,(Variable_Count*3)-2,3)	
197		= OFFSET(Var_Ref,(Variable_Count*3)-2,4)	
198		= OFFSET(Var_Ref,(Variable_Count*3)-2,5)	
199		= (B198+(4*B197)+B198)/6	
200		= ABS((B196-B198)/6)	
201		= NORMINV(B195,B199,B200)	
202		= FORMULA(B201,ABSREF(OFFSET(Var_Ref,(Variable_Count*3-1),2),!$A$1))	
203		= END.IF()	
204		= IF(Distribution_Type=3)	
205		= RAND()	
206		= OFFSET(Var_Ref,(Variable_Count*3)-2,3)	
207		= OFFSET(Var_Ref,(Variable_Count*3)-2,4)	
208		= NORMINV(B205,B206,B207)	
209		= FORMULA(B208,ABSREF(OFFSET(Var_Ref,(Variable_Count*3-1),2),!$A$1))	
210		= ELSE.IF(Distribution_Type=4)	
211		= RAND()	
212		= OFFSET(Var_Ref,(Variable_Count*3-2),3)	
213		= IF(B211>OFFSET(Var_Ref,(Variable_Count*3),4),OFFSET(Var_Ref,(Variable_Count*3-2),5),B$212)	
214		= IF(B211>OFFSET(Var_Ref,(Variable_Count*3),5),OFFSET(Var_Ref,(Variable_Count*3-2),6),B$212)	
215		= IF(B211>OFFSET(Var_Ref,(Variable_Count*3),6),OFFSET(Var_Ref,(Variable_Count*3-2),7),B$212)	
216		= IF(B211>OFFSET(Var_Ref,(Variable_Count*3),7),OFFSET(Var_Ref,(Variable_Count*3-2),8),B$212)	
217		= IF(B211>OFFSET(Var_Ref,(Variable_Count*3),8),OFFSET(Var_Ref,(Variable_Count*3-2),9),B$212)	
218		= IF(B211>OFFSET(Var_Ref,(Variable_Count*3),9),OFFSET(Var_Ref,(Variable_Count*3-2),10),B$212)	
219		= IF(B211>OFFSET(Var_Ref,(Variable_Count*3),10),OFFSET(Var_Ref,(Variable_Count*3-2),11),B$212)	
220		= MAX(B213:B219)	
221		= FORMULA(B220,ABSREF(OFFSET(Var_Ref,(Variable_Count*3-1),2),!$A$1))	
222		= END.IF()	
223		= NEXT()	
224			
225		= IF(Objective_Reference="")	Establish Link to Worksheet
226	!Champus_Total	= FORMULA(INDEX(NAMES(),,Objective_Name),A226)	
227		= REPLACE(A226,1,0,"!")	
228		= FORMULA(B227,A226)	
229		= GET.NAME(A226)	
230		= FORMULA.GOTO(GET.CELL(5,B229))	
231		= REFTEXT(ACTIVE.CELL(),FALSE)	
232		= ABSREF(B231,!$A$1)	
233		= SET.VALUE(Present_Objective,B232)	
234		= ELSE()	
235		= ABSREF(Objective_Reference,!$A$1)	
236		= SET.VALUE(Present_Objective,B235)	
237		= END.IF()	
238	Present_Objective	= Present_Objective	Record Objective Value
239		= IF(Present_Objective<Lower_Limit)	

	A	B	C
1	names	Commands	comments
2			
240		= FORM JLA(OFFSET(Output_Ref,2,0)+1,OFFSET(Output_Ref,2,0))	
241		= END IF()	
242		= FOR("Count",2,Number_to_Display+2)	
243		= IF(Present_Objective<OFFSET(Output_Ref,Count+1,1))	
244		= IF(Present_Objective>OFFSET(Output_Ref,Count,1))	
245		= FORMULA(OFFSET(Output_Ref,Count,0)+1,OFFSET(Output_Ref,Count,0))	
246		= END IF()	
247		= END IF()	
248		= NEXT()	
249		= IF(Present_Objective>Upper_Limit)	
250		= FORMULA(OFFSET(Output_Ref,Number_to_Display+1,0)+1,OFFSET(Output_Ref,Number_to_Display+2,0))	
251		= END I'()	
252			
253	Total_Objective	= Total_Objective+Present_Objective	Sum Objective
254	Total_Squared	= Total_Squared+(Present_Objective^2)	
255			
256		=NEXT()	
257			
258			
259			
260		=FOR("Count",2,Number_to_Display+2)	Calculate Relative Frequency
261		= FORMULA(OFFSET(Output_Ref,Count,0)/Number_Iterations,OFFSET(Output_Ref,Count,2))	
262		=NEXT()	
263			
264		=FORMULA(OFFSET(Output_Ref,2,2),OFFSET(Output_Ref,2,3))	Calculate Cumulative Frequen
265		=FORMULA(OFFSET(Output_Ref,2,0),OFFSET(Output_Ref,2,4))	
266		=FOR("Count",3,Number_to_Display+2)	
267		= FORMULA(OFFSET(Output_Ref,Count,2)+OFFSET(Output_Ref,Count-1,3),OFFSET(Output_Ref,Count,3))	
268		= FORMULA(OFFSET(Output_Ref,Count,0)+OFFSET(Output_Ref,Count-1,4),OFFSET(Output_Ref,Count,4))	
269		=NEXT()	
270			
271		=FORMULA(Total_Objective/Number_Iterations,OFFSET(Output_Ref,2,5))	Record Average and Std Dev
272		=SQRT(((Number_Iterations*Total_Squared)-(Total_Objective^2))/(Number_Iterations^2))	& Most Likely
273		=FORMULA(B272,OFFSET(Output_Ref,2,6))	
274		=SET NAME("Freq_Output",AC3:OFFSET(Output_Ref,Number_to_Display+2,2))	
275		=MATCH(MAX(Freq_Output),Freq_Output,0)	
276		=FORMULA(OFFSET(Output_Ref,B275+1,1),AG6)	
277			
278		=FOR("Count",1,Number_Variables)	Return Variables to Original
279		= FORMULA(OFFSET(Var_Ref,(Count*3-2),2),ABSREF(OFFSET(Var_Ref,(Count*3-1),2),'$A$1))	Values
280		=NEXT()	
281			
282		=FORMULA(Output_Range,K261)	Ask to Transfer Data
283		=DIALOG BOX(Input_Three)	
284		=IF(B283=FALSE,GOTO(B288))	
285		=COPY(AA2:OFFSET(Output_Ref,Number_to_Display+3,6),ABSREF(K261,!$A$1))	
286			
287			
288		=DIALOG BOX(Input_Four)	Ask to Record Input
289		=IF(B288=FALSE,GOTO(B294))	
290		=COPY(N2:OFFSET(Var_Ref,60*3,11),ABSREF(K275,'$A$1))	
291		=COPY(Transfer_1,ABSREF(RELREF(OFFSET(ABSREF(K275,'$A$1),0,12),'$A$1),'$A$1))	
292		=COPY(Transfer_2,ABSREF(RELREF(OFFSET(ABSREF(K275,'$A$1),0,13),'$A$1),'$A$1))	
293			
294		=ALERT("Simulation Complete",3)	Simulation Complete
295		=MESSAGE(FALSE,)	
296			
297		=RETURN()	

	E	F	G	H	I	J	K	L
1	DIALOGS							
2	type	x	y	wide	high	text	init/result	names
3	Input One							
4				522	420	Objective & Simulation Parameters		
5	1	414	14	88		OK		
6	13	18	7			First Iteration to Clear Data	FALSE	First_Iteration
7	13					Variable Values Previously Entered	TRUE	Variables_Known
8	13					Allow Screen Updating (S.ows Simu	FALSE	Screen_Update
9	13					Recall Input Data from Worksheet	FALSE	Recall_Input
10	5	44	80			Input Range		
11	10	168	77	154			R4C42	Input_Range
12	14	19	103	316	96	Objective Variable		
13	5	28	127			Reference		
14	10	139	126	160				Objective_Reference
15	5	33	148			OR		
16	5	30	170			Name		
17	21	107	169	190	108	NAMES()	1	Objective_Name
18	5	30	210			Number of Variables for Monte Carl		
19	7	404	230	96			54	Number_Variables
20	14	20	254	483	136	Simulation Parameters		
21	5	44	273			Number of Iterations		
22	7	326	272	160			500	Number_Iterations
23	5	43	296			Lower Limit		
24	8	325	295	160			400000	Lower_Limit
25	5	42	321			Upper Limit		
26	8	325	317	160			900000	Upper_Limit
27	5	41	342			Number of Lines to Display		
28	7	325	340	160			25	Number_to_Display
29	5	42	364			Output Range		
30	10	325	364	160			R4C34	Output_Range
31	2	414	43	88		Cancel		

	E	F	G	H	I	J	K	L
1	DIALOGS							
2	type	x	y	wide	high	text	init/result	names
34	Input Two							
35				842	426	Variable Entry		
36	1	738	22	88		DONE		
37	14	16	40	317	164	Variable Information		
38	5	25	64			Reference		
39	10	136	63	160			R24C31	Variable_Reference
40	5	30	85			OR		
41	5	25	107			Name		
42	21	104	106	190	108	NAMES()	#N/A	Variable_Name
43	5	25	134			Type of Distribution		
44	11						1	Distribution_Type
45	12	193	132			Uniform		
46	12					Triangular		
47	12					Normal		
48	12					Table		
49	1	40	160	88		OK		
50	214	16	214	320	90	Uniform Distribution Information		
51	205	37	243			Low		
52	8	138	243	160			320000	
53	205	37	273			High		
54	8	138	273	160			375000	
55	214	17	314	320	96	Triangular Distribution Information		
56	205	26	336			Pessimistic		
57	8	138	334	160			0.15	
58	205	26	359			Most Likely		
59	8	138	357	160			0.07	
60	205	26	385			Optimistic		
61	8	138	383	160			0.03	
62	214	355	5	357	84	Normal Distribution Information		
63	205	366	25			Average		
64	8	530	24	160			1	
65	205	366	51			Standard Deviation		
66	8	530	49	160			0.5	
67	214	355	101	469	312	Table Distribution Information		
68	205	391	147			1		
69	205	391	170			2		
70	205	391	194			3		
71	205	391	223			4		
72	205	391	252			5		
73	205	391	275			6		
74	205	391	302			7		
75	205	391	329			8		
76	205	391	357			9		
77	205	519	125			Value		
78	205	686	122			Frequency		
79	8	460	143	160			0.13	
80	8	640	143	160			20	
81	8	460	170	160			0.135	
82	8	640	170	160			15	
83	8	460	195	160			0.14	
84	8	640	195	160			10	
85	8	460	221	160			0.145	
86	8	640	221	160			10	
87	8	460	245	160			0.15	
88	8	640	245	160			10	
89	8	460	272	160			0.155	
90	8	640	272	160			10	
91	8	460	297	160			0.16	
92	8	640	297	160			10	
93	8	460	322	160			0.165	
94	8	640	322	160			5	
95	8	460	351	160			0.17	
96	8	640	351	160			5	
97	5	26	15			Variable Number		
98	7	209	9	96			48	
99	2	738	49	88		Cancel		

LIST OF REFERENCES

1. Barr, Chritopher, "The Next Crop of Spreadsheets will Play "What If"," *PC Magazine*, v. 11, n. 7, p.29, 14 April 1992.

2. Blackstone, Jr., John H., and Watson, Hugh J., *Computer Simulation*, 2d ed., John Wiley and Sons, 1981.

3. Schmidt, J. W., and Taylor, R.E., *Simulation and Analysis of Industrial Systems*, p. 327, Richard D. Irwin, Inc., 1970.

4. Maassen, Leland, and Whipple, David, *The Champus Budgeting and Programming Process*, pp. 9-28, Master's Thesis, Naval Postgraduate School, Monterey, CA, September, 1975.

BIBLIOGRAPHY

Bhaskar, Krish N., *Building Financial Models*, Associated Business Programmes, 1978.

Brandy D. Brent, and Gross, James R., "A Demonstration of Alternative Approaches for and Inventory Simulations Model," paper presented at the Society for Computer Simulation Multiconference on Simulation in Business and Management, Anaheim, California, 23-25 January 1991.

Gordon, Geoffrey, *System Simulation*, Prentice-Hall, Inc. 1969.

House, William C., *Business Simulation for Decision Making*, Petrocelli, 1977.

Meier, Robert C., Newell, William T., and Pazer, Harold, *Simulation in Business and Economics*, Prentice-Hall, Inc., 1969.

Mould, Gill, "Spreadsheet Simulation," paper presented at the Society for Computer Simulation Multiconference on Simulation in Business and Management, San Diego, California, 17-19 January 1990.

Nathan, Dinakar L., "A Focused Simulation Approach to Justify Manufacturing Modernization Strategic Plans," paper presented at the Society for Computer Simulation Multiconference on Simulation in Business and Management, San Diego, California, 17-19 January 1990.

Nersesian Roy L., "Application/Implementation of Simulation to Financing," paper presented at the Society for Computer Simulation Multiconference on Simulation in Business and Management, San Diego, California, 17-19 January 1990.

<u>Software</u>

Microsoft Windows 3.1, Microsoft Corporation, Redmond, Washington.

Microsoft Excel 4.0 for Windows, Microsoft Corporation, Redmond, Washington.

INITIAL DISTRIBUTION LIST

1. Defense Technical Information Center 2
Cameron Station
Alexandria, VA 22304-6145

2. Library, Code 52 2
Naval Postgraduate School
Monterey, CA 93943-5002

3. Professor Shu Sheng Liao, Code AS/Lc 2
Department of Adminstrative Science
Naval Postgraduate School
Monterey, CA 93943-5000

4. RADM Richard D. Milligan, USN (Ret.), Code AS/Ml 2
Department of Adminstrative Science
Naval Postgraduate School
Monterey, CA 93943-5000

4. LT Dixon K. Hicks 2
c/o J. B. Torbit
952 Midway Drive
Alpine, CA 91901